НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Факультет інформатики та обчислювальної техніки

Кафедра автоматики та управління в технічних системах

«На правах рукопису»
УДК 004.62

До захисту допущено:
Завідувач кафедри
_____ Олександр РОЛІК
«___»_____20__ р.

Магістерська дисертація

на здобуття ступеня магістра
за освітньо-професійною програмою «Інтегровані інформаційні системи»
зі спеціальності 126 «Інформаційні системи та технології»
на тему: «Система аналізу бізнес-даних з використанням технології оброблення природних мов»

Виконав (-ла):
студент (-ка) VI курсу, групи ІА-з92мп
Прокопенко Микита Ігорович _____

Керівник:
доцент каф. АУТС, к.т.н., доцент,
Букасов Максим Михайлович _____

Рецензент:

_____

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент (-ка) _____

Київ – 2020 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра автоматики та управління в технічних системах

Рівень вищої освіти – другий (магістерський)

Спеціальність – 126 «Інформаційні системи та технології»

Освітньо-професійна програма «Інтегровані інформаційні системи»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Олександр РОЛІК
«___»_____20__ р.

ЗАВДАННЯ
на магістерську дисертацію студенту
Прокопенку Микиті Ігоровичу

1. Тема дисертації «Система аналізу бізнес-даних з використанням технології оброблення природних мов», науковий керівник дисертації Букасов Максим Михайлович, к.т.н., доц., затверджені наказом по університету від «26» 10 2020 р. №3132-с

2. Термін подання студентом дисертації _____

3. Об’єкт дослідження: системи аналізу бізнес-даних

4. Вихідні дані: система аналізу бізнес-даних з використанням технології оброблення природних мов

5. Перелік завдань, які потрібно розробити: дослідження ринку BI платформ та з’ясування, які засоби вже імплементовано для спрощення сценаріїв використання; аналіз основних задач та проблем обробки природних мов, огляд найсучасніших інструментів з обробки природних мов; розроблення діаграми сценаріїв використання системи; проектування архітектури програмного рішення та розроблення структурної схеми; вибір технологій та розробка системи; проведення тестування на основі реального набору даних, розроблення стартап-проекту.

6. Орієнтовний перелік графічного (ілюстративного) матеріалу: діаграма сценаріїв використання, структурна схема, схема бази даних, діаграма класів модулю роботи із джерелами даних, діаграма класів модулю роботи зі сховищами даних, діаграма

класів модулю представлення даних, блок-схема алгоритму оброблення запиту природною мовою, діаграма розготрання.

7. Орієнтовний перелік публікацій:

8. Дата видачі завдання 04.09.2020

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації	Примітка
1.	Отримання завдання	04.09.2020	
2.	Дослідження ринку BI платформ та з'ясування, які засоби вже імплементовано для спрощення сценаріїв використання	04.09.2020-19.09.2020	
3.	Аналіз основних задач та проблем обробки природних мов, огляд найсучасніших інструментів з обробки природних мов	19.09.2020-08.10.2020	
4.	Розроблення діаграми сценаріїв використання системи	08.10.2020-16.10.20	
5.	Проектування архітектури програмного рішення та розроблення структурної схеми	16.10.2020-24.11.2020	
6.	Вибір технологій та розробка системи	24.11.2020-18.11.2020	
7.	Проведення тестування на основі реального набору даних	18.11.2020-24.11.2020	
8.	Розроблення стартап-проекту	24.11.2020-26.12.2020	
9.	Оформлення дисертації	26.11.2020-01.12.2020	
10.	Подання дисертації до попереднього захисту	01.12.2020	

Студент Микита ПРОКОПЕНКО

Науковий керівник Максим БУКАСОВ

РЕФЕРАТ

Магістерська дисертація на здобуття ступеня «магістр» за освітньо-професійною програмою підготовки «Інтегровані інформаційні системи» на тему «Система аналізу бізнес-даних з використанням технології оброблення природних мов». Дисертація містить 101 сторінок, 43 рисунки, 46 таблиць, 8 додатків, 17 джерел.

Актуальність. Системи аналізу бізнес-даних є складними для сприйняття звичайним користувачем, адже інтерфейс взаємодії є громіздким та «навантаженим». Останнім часом спостерігається тенденція нестачі спеціалістів по даних, які мають достатні технічні знання для роботи з BI системами. Тому на базі цього постала проблема спрощення взаємодії та ліквідації бар'єру між будь-яким користувачем та системою бізнес-аналізу. У зв'язку з цим було запропоновано новий шлях у розвитку BI – використання технології оброблення природних мов. Проте поки цей підхід інтегровано як опціональне рішення.

Метою магістерської дисертації є спрощення взаємодії користувача із програмними засобами бізнес-аналітики за допомогою інструментів оброблення природних мов.

Об'єкт дослідження – системи аналізу бізнес-даних, предмет дослідження – спосіб взаємодії користувача з BI системою.

На базі огляду сучасних рішень на ринку BI платформ та аналізу засобів, які вже імплементовано для спрощення сценаріїв використання, розроблено систему аналізу бізнес-даних з використанням технології оброблення природних мов. Така система призначена для впровадження у діяльність компаній середнього та великого бізнесу, які мають необхідність у BI, але відчувають нестачу спеціалістів по даним.

Апробація результатів дисертації. Результати магістерської дисертації впроваджені та використовуються у діяльності підприємства ТОВ «Девелопекс».

Ключові слова: БІЗНЕС-АНАЛІТИКА, ОБРОБКА ПРИРОДНИХ МОВ, АНАЛІЗ БІЗНЕС-ДАНИХ, ЧАТБОТ, ІНТЕГРАЦІЯ ДАНИХ, ПРЕДСТАВЛЕННЯ ДАНИХ.

ABSTRACT

Master's dissertation on the educational-professional level training program “Master” on the theme “Business data analysis system with natural language processing technology”. The dissertation contains 101 pages, 43 figures, 46 tables, 8 applications, 17 sources.

Relevance. Business data analysis systems are difficult for the average user to perceive, because the interface is cumbersome and “loaded”. Lately, there has been a shortage of specialists in data who have sufficient technical knowledge to work with BI systems. Therefore, the problem of simplification of interaction and elimination of the barrier between any user and the business analysis system arose on this basis. In this regard, a new way in the development of BI was proposed – the use of natural language processing technology. However, so far this approach has been integrated as an optional solution.

The purpose of the master's dissertation is to simplify user interaction with business intelligence software with natural language processing tools.

The object of research is business data analysis system, the subject of research – the way the user interacts with the BI system.

Based on a review of current solutions in the market of BI platforms and analysis of tools that have already been implemented to simplify usage scenarios, a system of business data analysis using natural language processing technology has been developed. Such a system is designed to be integrated into the activities of medium and large businesses that need BI, but do not have a sufficient number of specialists in data.

Approbation of dissertation results. The results of the master's dissertation are implemented and used in the activities of the company “Developex”.

Keywords: BUSINESS ANALYTICS, NATURAL LANGUAGE PROCESSING, BUSINESS DATA ANALYSIS, CHATBOT, DATA INTEGRATION, DATA PRESENTATION.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

БД – база даних

ООП – об’єктно-орієнтоване програмування

СУБД – система управління базами даних

API – Application Programming Interface – інтерфейс прикладного програмування

BI – Business Intelligence – бізнес-аналітика

DAO – Data Access Object – шаблон проектування

DI – Dependency injection – впровадження залежності

DTO – Data Transfer Object – об’єктом передачі даних

ETL – Extract Transform-Load

HTTP – HyperText Transfer Protocol – протокол передачі гіпертексту

JDBC – стандарт взаємодії Java з базами даних

JSON – JavaScript Object Notation – нотація об'єкта javascript

KPI – Key Performance Indicator – показник досягнення успіху

MVC – Model-View-Controller – архітектурний шаблон

NLP – Natural Language Processing – технологія оброблення природних мов

ORM – Object-Relational Mapping – об’єктно-реляційне відображення

POS – Parts of Speech – частини мови

URL – Uniform Resource Locator – стандартизована адреса ресурсу

ВСТУП

У сучасних умовах цифрової економіки підприємства великого та середнього бізнесу все частіше стикаються з необхідністю перегляду своїх підходів до управління та організації процесів. Основний фокус трансформацій – впровадження і практичне використання цифрових технологій збору, обробки, збереження, перетворення та передачі даних у будь-якій сфері діяльності. Підприємства, які орієнтуються на такий підхід отримують ряд конкурентних переваг: ефективне використання ресурсів, підґрунтя для правильних рішень в управлінні та виборі стратегії, розуміння потреб цільової аудиторії та потенційних клієнтів, можливість впровадження інноваційних рішень, які будуть мати попит, гнучкість та пристосовуваність до мінливих умов ринку. У зв’язку з цим, збільшується попит на використання систем класу Business Intelligence (BI) як засобів підвищення конкурентоспроможності та базису для прийняття ефективних рішень в управлінні.

Системи аналізу бізнес-даних є складними для сприйняття звичайним користувачем, адже інтерфейс взаємодії є громіздким та «навантаженим». Останнім часом спостерігається тенденція випередження пропозиції попитом на спеціалістів та менеджерів по даних, які мають достатні технічні знання для роботи з BI системами. Тому на базі цього постала проблема спрощення взаємодії та ліквідації бар’єру між будь-яким користувачем та системою бізнес-аналізу. На початку 2020 року було запропоновано новий шлях у розвитку BI – використання технології обробки природних мов. Цю ідею взяли до уваги лідери ринку, проте не трансформували свої продукти кардинально, а інтегрували новий підхід як опціональне рішення. Тому до цього часу створення системи бізнес аналізу з використанням обробки природних мов, яка є повністю орієнтованою на звичайного користувача залишається актуальним питанням.

Метою магістерської дисертації є спрощення взаємодії користувача із програмними засобами бізнес-аналітики за допомогою інструментів оброблення природних мов.

Задачі, які було вирішено для поставленої мети:

– аналіз основних задач та проблем обробки природних мов;

– дослідження ринку BI платформ та з’ясування, які засоби вже імплементовано для спрощення сценаріїв використання;

– огляд найсучасніших інструментів з обробки природних мов та визначення, які найбільш підходять для досягнення поставленої мети;

– аналіз задач обробки природних мов, які необхідно вирішити для реалізації системи;

– проектування архітектури програмного рішення;

– розробка системи, яка відповідає встановленим вимогам;

– проведення тестування на основі реального набору даних.

Об'єктом дослідження є системи аналізу бізнес-даних.

Предмет дослідження – спосіб взаємодії користувача з BI системою.

Апробація результатів дисертації. Результати магістерської дисертації були впроваджені та використовуються у діяльності ТОВ «Девелопекс».

1 ОГЛЯД ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Системи аналізу бізнес-даних

Системи бізнес-аналізу – вид інформаційних систем, які дозволяють трансформувати накопичені фактичні дані в корисні знання, що можуть бути використані для вирішення широкого спектру задач в області планування, виконання, формування звітності і моніторингу ефективності діяльності. В таблиці 1.1 наведено найпоширеніші задачі в різних аспектах діяльності підприємства, для вирішення яких використовуються сучасні системи BI [1].

Таблиця 1.1 – Найпоширеніші задачі, для вирішення яких використовують BI-системи

Аспект	Задачі
Топ-менеджмент	Виявлення збиткових і прибуткових напрямків діяльності організації, ґрунтуючись на аналізі великого обсягу внутрішньої і зовнішньої інформації. Оперативне прийняття стратегічних і тактичних рішень на основі постійного моніторингу поточного бізнесу. Своєчасне виявлення потенційних проблем в діяльності організації. Оцінка ефективності використання ресурсів, в тому числі дочірніми підприємствами.
Продаж	Планування продажів, оцінка виконання планів в режимі реального часу. Аналіз діяльності менеджерів з продажу, розрахунок бонусів. Аналіз продажів в різних областях, моніторинг динаміки продажів.

	Виявлення точок зростання в залежності від сезонних, тимчасових і інших чинників.
Фінанси	Планування бюджетів. Аналіз руху грошових коштів. Аналіз історії платіжної дисципліни кредиторів. Консолідація фінансової звітності.
Маркетинг	Аналіз конкуренції, рушійних сил в галузі, різних сегментів ринку. Маркетингові дослідження, виявлення потенційних покупців для нових товарів і послуг, прогнозування обсягу попиту і пропозиції. Аналіз ефективності маркетингових акцій. Виявлення найбільш ефективних інструментів просування товарів.
Виробництво і логістика	Формування оптимального виробничого плану. Оперативне управління поставками продукції, аналіз маршрутів доставки. Аналіз управління запасами на складах, контроль залишків. Моніторинг і контроль своєчасного відвантаження товарів, виявлення «вузьких місць», відстеження термінів доставки товарів, аналіз постачальників.

Проаналізувавши вищенаведені задачі, можна зробити висновок, що BI системи є ланкою, яка слугує для певної інтеграції функцій та технологій усіх відділів підприємства. До того ж об’єднує дані з будь-яких «точок» організації, баз даних не залежно від їх формату та каналу надання. Інформаційний потік може також виходити і за межі організаційних кордонів, обчислювальних платформ та спеціалізованих інструментів. Загальний принцип роботи систем аналізу бізнес даних наведений на рисунку 1.1.

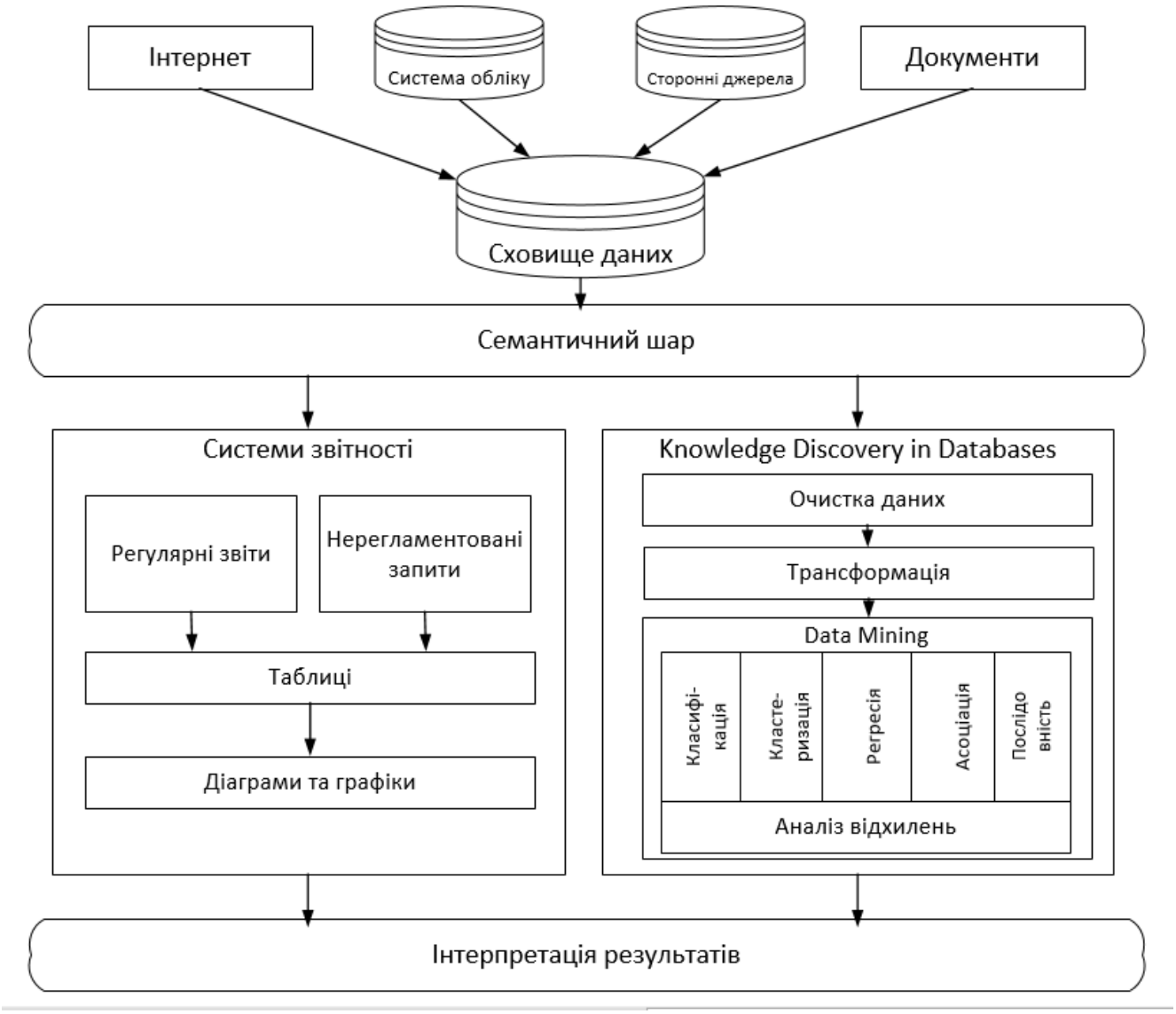

Рисунок 1.1 – Загальний принцип роботи систем аналізу бізнес-даних

Основний функціонал BI систем включає:

– збір даних з різних джерел, їх структуризація та зберігання в єдиній цілісній системі;

– аналіз великих об’ємів даних для формування і підтвердження гіпотез або розробки бізнес-рішень, спираючись на аналітику;

– моделювання можливих рішень для оцінки їх впливу на кінцеві показники діяльності і прогнозування подальшого розвитку на основі поточних даних;

– формування оперативної і стратегічної звітності, в тому числі сповіщення про відхилення показників від допустимих норм;

– збереження і систематизація знань з ціллю майбутньої передачі новим працівникам, для передавання досвіду і стабільного підвищення якості роботи.

Не дивлячись на такі масштабні можливості систем бізнес-аналізу, впровадження їх в роботу підприємств не гарантує миттєвого підвищення ефективності. Адже тут важливу роль грає людський фактор – професійність і обізнаність людей, які відповідальні за роботу з системою (зазвичай бізнес-менеджери й аналітики), та як вони тлумачать надані результати. Адже інтерфейс, інтерпретація результатів та в цілому взаємодія з системою є досить складними для пересічного користувача та інтуїтивно не до кінця зрозумілими. Тому коли постала така проблема нестачі спеціалістів в області даних, дослідники та розробники почали шукати можливі шляхи до спрощення взаємодії та звернули увагу на технології обробки природних мов (NLP).

1.2 Технології обробки природних мов

Ідея застосування програмування для обробки природних мов виникла в 1954 році, коли було проведено так званий Джорджтаунський експеримент (в Нью-Йорці, в штаб-квартирі IBM). В ході експерименту було продемонстровано автоматичний переклад біля шістдесяти речень з російської на англійську мову. У його основі була досить проста система – 6 граматичних правил та словник з 250 записів, а в якості предметної області – органічна хімія. Саме цей експеримент позитивно вплинув на розвиток машинного перекладу та NLP в найближчі 10-12 років. І вже у 1964 році американський спеціаліст з інформатики Джозеф Вейценбаум розробив перший у світі чатбот «Eliza». Користувач міг ввести деяке твердження (або декілька) на природній мові і отримував доречну відповідь від машини. Це була перша програма, яка робила можливою певного роду бесіду між людиною та комп’ютером на природній мові.

У 1970 році була розроблена програма, що розуміла природну мову, в якій користувач надавав певні інструкції з переміщення різних блоків по-різному, тобто в залежності від контексту.

У 1978 році розроблена система NLP з базами даних, особливістю якої стало використання семантичної граматики для аналізу питань та запитів до розподіленої БД.

У 1997 році – запуск першої пошукової системи з функцією введення питань на природній мові, 2006 рік – презентація IBM Watson – суперкомп'ютера, який поєднав у собі штучний інтелект та аналітичне програмне забезпечення як «машина для відповідей на питання» [2].

У 2013 році Google оновлює свій алгоритм пошуку, щоб вийти за рамки пошукового запиту і сконцентруватися на намірах користувача.

2011 рік – по сьогоднішній день – ера віртуальних помічників (Siri, Alexa, Google Assistant), які стають стандартним інструментом NLP у більшості інтелектуальних пристроях.

У 2020 році виникає нова тенденція у сфері бізнесу – застосування обробки природних мов в системах аналізу бізнес даних з метою спрощення отримання доступу до складної бізнес-інформації.

Обробка природної мови – це аналіз лінгвістичних даних (найчастіше у формі текстових) з використанням обчислювальних методів. Метою обробки природної мови є, як правило, побудова подання тексту, який додає структуру до неструктурованої природної мови, використовуючи лінгвістичні форми. Вона може мати синтаксичний характер – граматичні зв'язки між складовими тексту, або більш семантичний – фіксація змісту, переданого текстом. Обробка природної мови, відноситься до технологій штучного інтелекту, яка дозволяє комп'ютерам розуміти, інтерпретувати і маніпулювати природною людською мовою.

Природна мова – складний інструмент, який має велику кількість граматичних правил, контекстних та семантичних залежностей, відтінкових нюансів, тому працювати з цим дуже складно. Адже навіть простий запит можна трактувати по-різному. А мета NLP не просто надати будь-яку інформацію, а таку, яка буде значущою і корисною. На сьогоднішній день задачі, для вирішення яких використовується ця технологія, можна розділити на чотири основні класи:

– розпізнавання та аналіз мови;

– синтез (генерація) мови;

– розпізнавання та аналіз тексту;

– синтез (генерація) тексту [3].

Така класифікація корисна для розуміння теоретичної бази, проте в практичному використанні вона значно розширилась задачами, які можна віднести хоча б до двох вищезгаданих класів. Найбільш поширені з них:

– інформаційний пошук (письмовий або усний);

– класифікація та кластеризація текстів (ціль: передбачати теги, категорії, тональність, застосування: фільтрація спаму, класифікація документів за переважним змістом);

– анотування та рефератування текстів (ціль: моделювання мови – пророкування наступного або попереднього слова (слів), генерація тексту, застосування: створення анотацій та рефератів до тексту);

– машинний переклад;

– вилучення цінних фактів та знань із тексту;

– розробка діалогових систем (типу «питання-відповідь») та чат-боти;

– голосові помічники (наприклад, автоматизована допомога покупцю при замовленні товарів та послуг тощо).

Для виконання вищенаведених задач в NLP зазвичай застосовують три основні групи підходів: на основі правил, «традиційне» машинне навчання та нейронні мережі [4].

Підходи, засновані на правилах використовуються найдовше, але до цих пір не втратили актуальності, адже є перевіреними і ефективними. Правила, що застосовуються до тексту, можуть дати багато інформації: з’ясувати до якої частини мови відноситься те чи інше слово, чи підходить його конструкція під певний шаблон тощо. Навчальні приклади засновані на даних підходах – регулярні вирази та контекстно-вільні граматики. Підходи на основі правил:

– зосереджені на зіставленні зі зразком або синтаксичному аналізі;

– часто можна розглядати як методи «заповнення прогалин»;

– мають високе відкликання, але невелику точність, у зв’язку з чим вони можуть мати високу продуктивність в конкретних випадках використання, але часто страждають від зниження продуктивності при узагальненні.

«Традиційні» підходи до машинного навчання включають імовірнісне моделювання, максимізацію ймовірності та лінійні класифікатори. Вони характеризуються:

– навчальні дані – у даному випадку корпус з розміткою;

– розробка функцій – тип слова, оточуючі слова, великі літери, множина тощо;

– навчання моделі за параметрами з подальшим підлаштування по тестовим даним;

– умовивід (застосування моделі до даних тесту), що характеризується пошуком найбільш вірогідних слів, наступного слова, найкращої категорії тощо;

– «семантичне заповнення слота».

Підходи, засновані на нейронних мережах, схожі на «традиційне» машинне навчання, але з такими відмінностями:

– проектування функцій зазвичай пропускається, через те, що мережі «навчаються» важливим функціям (зазвичай це одна із найбільших переваг використання нейронних мереж для NLP);

– потоки необроблених параметрів без спеціальних функцій подаються в нейронні мережі;

– дуже великий тренувальний корпус.

Найпоширенішими нейронними мережами, що використовуються в NLP є рекурентні та згорткові.

З самого початку дослідники області обробки природних мов стикнулися з великою кількістю проблем, вирішення яких знаходиться в постійному прогресі. На сьогоднішній день повністю закрито багато спірних моментів та складнощів, проте все залишаються задачі, які складно вирішити і які залишаються відкритими протягом декількох десятиліть. Якщо розглянути детальніше такі задачі, то можна дійти висновку, що усі проблеми, пов’язані з ними, схожі і діляться на дві групи: стосуються даних та розуміння (таблиця 1.2) [5].

Таблиця 1.2 – Проблеми обробки природних мов

Група	Проблема	Опис проблеми
	Мови з низьким рівнем ресурсів	Проблема полягає в тому, що, хоча існує маса даних по популярним мовам, таким як англійська чи китайська, існують тисячі мов, якими розмовляють дуже мало людей, і, отже, їм приділяється набагато менше уваги. Тільки в Африці налічується 1250-2100 мов, але даних по ним майже немає. Крім того, дуже складно переносити завдання, що вимагають реального розуміння природної мови, з мов з великими ресурсами на мови з низьким рівнем ресурсів.
	Великий документ (або декілька)	Проблема пов'язана з великими або множинними документами, оскільки поточні моделі в основному засновані на повторюваних нейронних мережах, які не можуть добре уявляти довші контексти. Робота з великими контекстами вимагає масштабування існуючих систем до тих пір, поки вони не зможуть читати цілі книги і сценарії.
	Оцінка мовних технологій	Проблемою оцінки мовних технологій, особливо такої складної, як діалог, часто нехтують, але це важливий момент: потрібні як поглиблені, так і ретельні дослідження, які проливають світло на те, чому певні підходи працюють, а інші ні, і

		розвивають оцінку міри на їх основі. Потрібне нове покоління наборів оціночних даних та завдань, які показують, чи насправді методи поширюються на усе різноманіття природних мов.
	Двозначність	Ця проблема є основною NLP – розуміння та моделювання елементів у змінному контексті. У природній мові слова є унікальними, але можуть мати різне значення залежно від контексту, що призводить до двозначності на лексичному, синтаксичному та семантичному рівнях. Для вирішення цієї проблеми NLP пропонує кілька методів, таких як оцінка контексту або введення позначок POS (Parts of Speech), проте розуміння семантичного значення слів у фразі залишається відкритим питанням.
	Синонімія	Ще одним явищем природних мов є той факт, що можна висловити одну і ту ж ідею різними термінами, які також залежать від конкретного контексту. У завданнях NLP необхідно включати знання синонімів та різні способи називання одного і того ж об’єкта чи явища, особливо коли мова йде про завдання високого рівня, що імітують людський діалог.

	Кореференція	Процес пошуку всіх виразів, що посилаються на одну й ту саму сутність у тексті, називається кореферентністю. Це важливий момент для багатьох завдань NLP вищого рівня, які передбачають розуміння природної мови (узагальнення документів, відповіді на запитання та витяг інформації).
	Особистість, намір, емоції і стиль	Залежно від особистості автора чи оратора, їхніх намірів та емоцій, вони також можуть використовувати різні стилі для вираження однієї і тієї ж ідеї.

1.3 Використання засобів обробки природних мов у бізнес-аналітиці

Кожна людина на Землі щосекунди створює близько 1,7 МБ даних, які представники бізнесу збирають та аналізують для кращого розуміння потреб сучасності та подальшого використання в маркетингових стратегіях, логістики запасів, можливості продажів і багато іншого. Можливість отримувати ключові метрики і складати звіти по ним на постійній основі або в міру необхідності може значно підвищити продуктивність, підвищити ефективність і стимулювати зростання. Але дані надходять в різних структурах з різних джерел. Збір, систематизація та складання звітів – непросте завдання.

Рішення бізнес-аналітики вимірює ключові бізнес-метрики і відображає їх у вигляді діаграм і графіків на інформаційній панелі. Рішення з можливостями спеціальної звітності дозволяють користувачеві переміщатися по цим показникам і глибше поглиблюватися в дані, щоб відповісти на конкретні питання.

Як зазначалося вище, не всі рішення бізнес-аналітики прості у використанні. Багато вимагають технічних знань, наприклад, як писати запити SQL. Для бізнесу це означає, що потрібні кваліфіковані співробітники, наприклад фахівці з даними, що

володіють технічними знаннями для створення звітів і аналізу даних. Але попит на фахівців з аналізу даних настільки високий, що пропозиція не встигає. Таким чином, на ринку бізнес-аналітики стався зсув в бік спрощення взаємодії з користувачем з упором на простоту, а саме на використання обробки природних мов. У більшості бізнес-аналітичних систем це реалізується за допомогою розробки діалогових систем (типу «питання-відповідь») або чат-боту. Такий підхід дозволяє користувачам виконувати пошук в даних так само, як вони використовували б пошукові системи, наприклад, такі як Google або Bing.

Принцип обробки природних мов в діалогових системах або чат-ботах є наступним (розгляд на прикладі запиту «How much do you tip a server?»). Першим кроком є поділ кожного слова на окремі сутності (токенізація) та визначення їхньої індивідуальної функції – частини мови в процесі, що носить назву POS тегування (рисунок 1.2) [6].

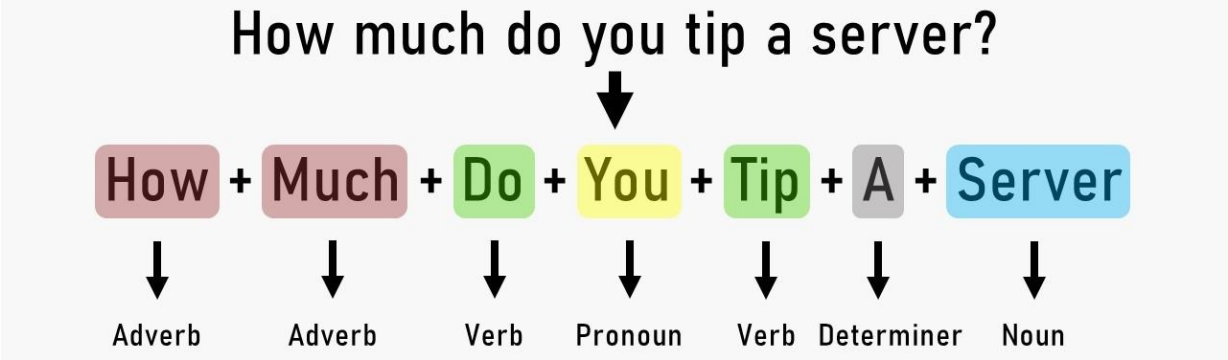

Рисунок 1.2 – POS-тегування

Другий крок – створення дерева синтаксичного аналізу. Це процес передбачає розбиття речення на фрази, щоб краще зрозуміти намір запиту (рисунок 1.3).

Третій крок передбачає семантику, або те, як на значення слова впливає його зв’язок з іншими словами в реченні. У даному прикладі «server» в англійській мові може означати «офіціант» і відноситись до загальновживаного словника, у другому випадку –«сервер» з області технічних знань. Програма NLP, завдяки можливостям машинного навчання, може знати, що «tip» («чайові») часто використовується в контексті готельно-ресторанного бізнесу, тому вона вибирає «офіціант» як найбільш

вірогідний сенс. Далі програмне забезпечення BI аналізує отриманий результат, надсилає запит базу даних і візуалізує відповідь в найбільш підходящому форматі.

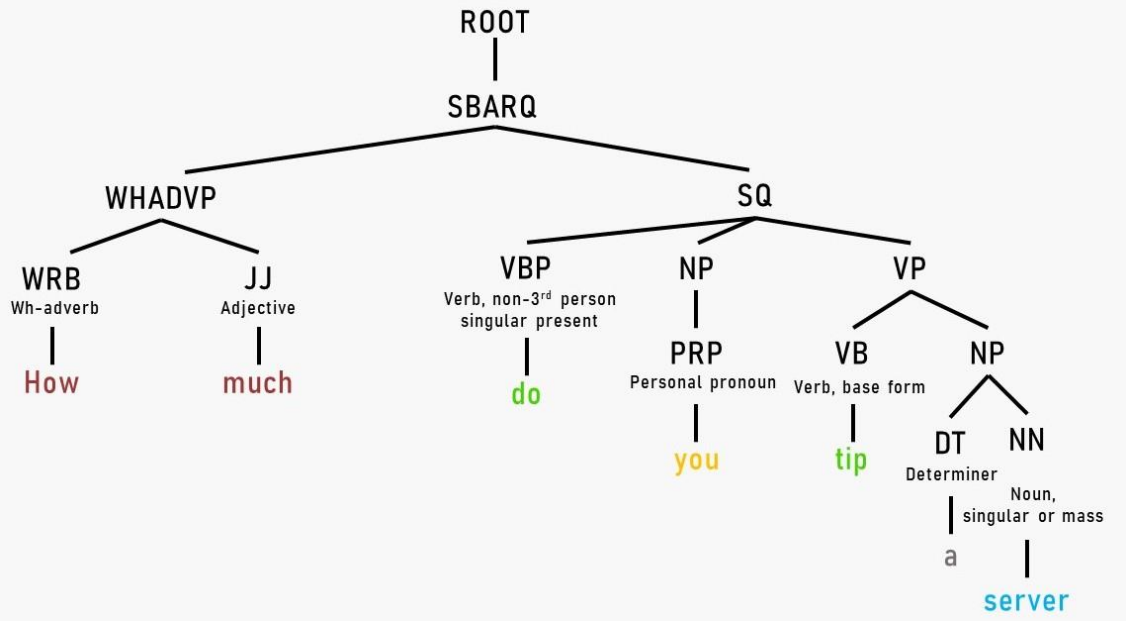

Рисунок 1.3 – Створення дерева синтаксичного аналізу

Використання такого підходу в бізнес-аналітиці має ряд переваг:

– демократизація даних: цінність запитів до даних за допомогою NLP полягає в можливості отримати доступ до складної бізнес-інформації, використовуючи один з перших придбаних навичок природну мову; вирішує проблему нестачі фахівців по роботі з даними;

– генерація корисних ідей: надаючи великій кількості користувачів можливість досліджувати дані і маніпулювати ними в пошуках цінних ідей, вони з більшою ймовірністю призведуть до прийняття більш ефективних бізнес-рішень;

– економія часу та ресурсів: компаніям більше не потрібно витрачати час і ресурси на те, щоб навчитися створювати специфічні запити до даних, адже обробка природної мови дає можливість кожному користувачеві, незалежно від технічних навичок, виконувати складні запити, просто задаючи питання природною мовою.

2 АНАЛІЗ ІСНУЮЧИХ РІШЕНЬ

Спираючись на дослідження міжнародної дослідницької і консалтингової компанії IDC (International Data Corporation), світовий ринок бізнес-аналітики і даних планомірно зростає: в 2017 році він досянув $ 122 млрд, у 2018 році – вже $ 130 млрд. У 2020 році аналітики прогнозують зростання обсягу ринку до $ 203 млрд. Зростання ринку бізнес-аналітики в кілька разів випереджає зростання ринку інформаційних технологій в цілому за рахунок постійно зростаючої потреби компаній в якісному і глибокому аналізі даних.

Для аналізу лідерів на ринку у будь-якій галузі використовують візуальне представлення положення різних продуктів і їх виробників, яке дозволяє оцінити їх можливості – магічний квандрант. В кінці 2019 року дослідницька компанія Gartner обнародувала новий магічний квандрант на ринку рішень бізнес-аналітики, на якому розташувала 20 найпопулярніших BI систем (рисунок 2.1) [7].

Рисунок 2.1 – Магічний квандрант BI систем 2019 року

Лідерами третій рік поспіль стали 3 вендора – Qlik, Tableau і Microsoft. Кожна система BI оцінюється за двома критеріями: абсциса – цілісність бачення (Completeness of vision), ордината – досконалість платформи (Ability to execute). Цілісність системи складається з сукупності характеристик, таких як: доступність маркетингової стратегії, інноваційність, доступність незалежно від територіального розташування, адаптивність під будь-які галузі господарської діяльності. Під досконалістю платформи мається на увазі конкурентоспроможність, гнучкість під запити ринку, технічний супровід і т.д.

Для побудови магічного квадранта проводять дослідження, на основі детального опитування респондентів, що використовують дані системи (кількість респондентів наведено в таблиці 2.1).

Таблиця 2.1 – Кількість респондентів

Система бізнес-аналітики	Кількість респондентів, шт.	Доля від загальної кількості респондентів
Qlik	262	19.8
Oracle	92	6.9
MicroStrategy	78	5.9
SAP	79	6.0
SAS	102	7.7

Крім того, в рамках дослідження були виявлені причини придбання продукту бізнес-аналізу (рисунок 2.2). За результатами найважливішим критерієм є функціональність – 51%, далі співвідношення «ціна-якість» (40%) та заключним критерієм стала простота у використанні для менеджерів (37 %). Саме на ці три пункти слід звертати особливу увагу при аналізі існуючих рішень та подальшій розробці власної системи.

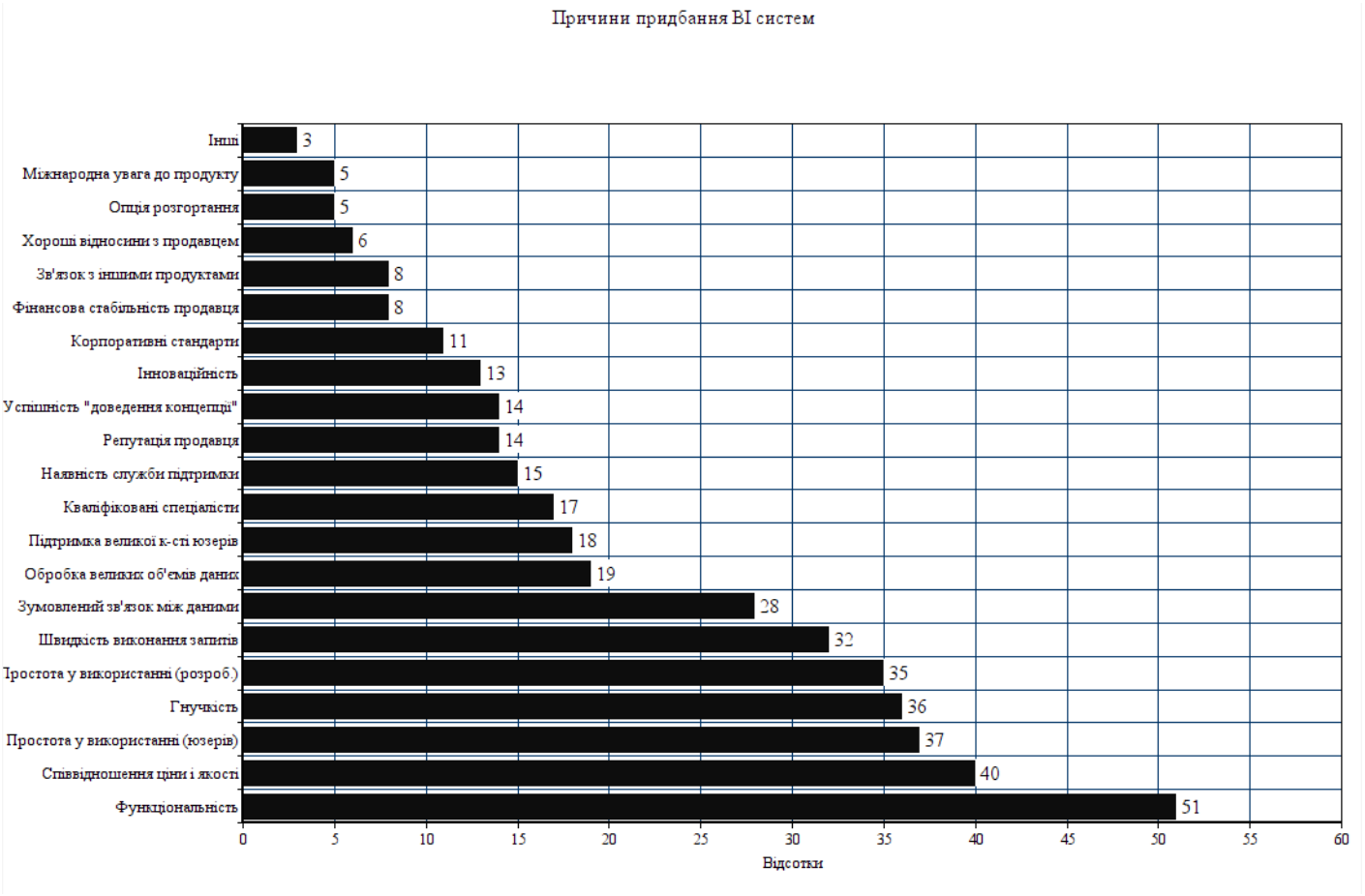

Рисунок 2.2 – Причини придбання BI систем

На сьогоднішній день існує велика кількість систем бізнес-аналізу, однак далеко не всі задовольняють потребам ринку. Кожна з систем має свої особливості, переваги і недоліки. Для їх порівняння проведено процес дослідження відібраних BI систем – лідерів ринку: QlikView, Tableau, та Power BI.

2.1 Система бізнес аналізу QlikView

Компанія QlikTech стала першовідкривачем у сфері бізнес аналізу, випустивши своє рішення Qlikview. Ця система має ряд особливостей. Наприклад, асоціативна модель даних: всі елементи взаємопов'язані по ключам, при фільтрації даних можна побачити взаємопов'язані елементи. Також присутня інтеграція будь-яких джерел даних: зручне підключення і об'єднання різнорідних джерел даних – від файлів Excel до ERP, CRM, WMS, а також Google Analytics, звітів Nielsen, GfK і т.д.

Усі розрахунки доступні «по кліку», можна отримати дані від показника по всій компанії/регіону до конкретної частини.

Права доступу до даних або об'єктів візуалізації налаштовуються аж до окремих рядків і стовпців, також є підтримка кластерної архітектури (QlikView Governance Dashboard), що дозволяє моніторити всі аспекти роботи серверу.

QlikView клієнт включає в себе потужний ETL-інструментарій (Extract Transform-Load), який має понад 200 вбудованих функцій для фільтрації, об'єднання і виконання складних операцій над даними для завантаження їх безпосередньо з різних джерел. Це можуть бути бази даних ERP, CRM або інших корпоративних систем, включаючи багатовимірні бази даних, спеціальні сховища, Excel або XML файли тощо. Крім цього, QlikView підтримує завантаження інформації веб-служб. Для побудови інтерфейсу додатку доступно більше 100 різних інтерактивних об'єктів, на кожен з яких можна натиснути для початку або продовження аналізу інформації [8]. В цілому, QlikView – це платформа, яка сконцентрована на користувачеві, на отримання ним даних. Переваги та недоліки даної BI системи наведені у таблиці 2.2.

Таблиця 2.2 – Переваги та недоліки QlikView

Переваги	Недоліки
Інтуїтивно зрозумілий інтерфейс	При фільтрації може скомбінувати декілька типів даних, коли це непотрібно
Досить легко фільтрувати дані при будь-якій візуалізації	Нема можливості об’єднувати результати в закладках
Швидкість створення графіків та таблиць	Користувач без технічних знать буде мати труднощі
Можливість відправляти звіти у зручному форматі	Неочевидний синтаксис
Підтримка імпорту даних з багатьох джерел	Деякі базові функції складно реалізувати
Зручність створення графіків, таблиць та	Складності у використанні в якості

фільтрів	інструменту для всієї компанії
Швидкість завантаження та обробки даних навіть при великих об’ємах	Незручність інтерфейсу для відправки звітів
Дозволяє спільне завантаження даних	

2.2 Система бізнес аналізу Tableau

Tableau – це інструмент візуалізації даних, який використовується в індустрії бізнес-аналітики, який допомагає спростити необроблені дані до зрозумілого формату. Він об’єднує дані, що зберігаються в різних джерелах, може збирати дані з будь-якої платформи: простої бази, як excel, складної, як Oracle, хмари, Amazon, Azure, Google Cloud SQL тощо. В Tableau аналіз даних здійснюється швидко, а візуалізації створюються у вигляді інформаційних панелей і робочих листів.

Пакет продуктів Tableau складається з наступних компонентів: Desktop, Public, Online, Server, Reader (рисунок 2.3) [9].

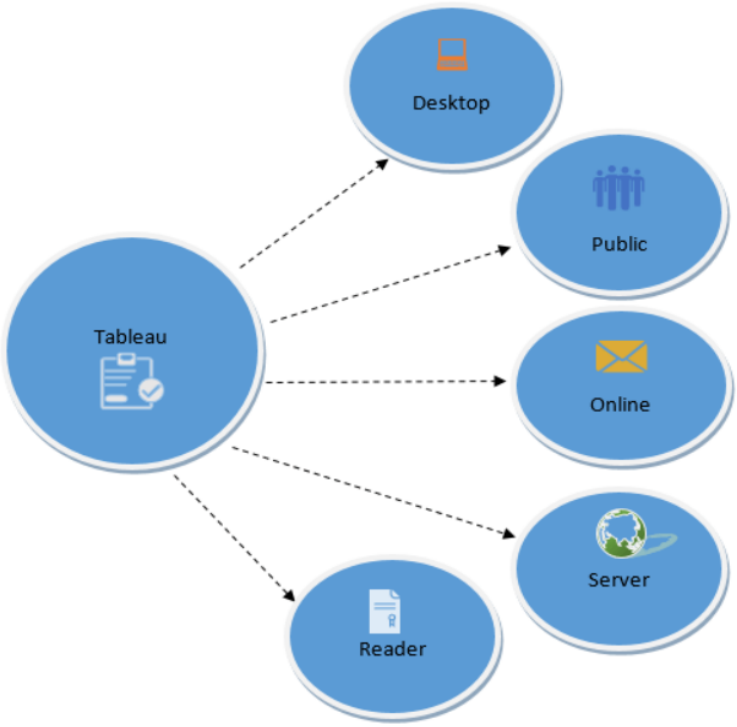

Рисунок 2.3 – Компоненти пакету продуктів Tableau

Tableau Desktop має багатий набір функцій і дозволяє кодувати і налаштовувати звіти. У цьому компоненті міститься весь необхідний інструментарій, починаючи зі створення діаграм, звітів і закінчуючи їх об'єднанням для формування панелі управління.

Для аналізу даних в реальному часі Tableau Desktop забезпечує підключення до сховища даних, а також до інших різних типів файлів. Створені тут звіти та інформаційні панелі можуть бути загально або локально доступними. Залежно від можливості підключення до джерел даних і можливості публікації Tableau Desktop підрозділяється на: Tableau Desktop Personal та Tableau Desktop Professional. Персональна версія зберігає звіт закритим та обмежує до нього доступ, проте його можна опублікувати в Інтернеті. Професійна версія надає повний доступ до всіх типів даних і дозволяє публікувати звіти на Tableau Server.

Tableau Public – компонент, який дозволяє зберігання звітів виключно в загальнодоступному хмарному сховищі системи, де доступ може отримати будь-хто з користувачів. Tableau Server – програмне забезпечення, спеціально створене для спільного використання звітів та візуалізацій по всій організації, даступ мають лише ліцензійні користувачі.

Tableau Online – інструмент для онлайн-обміну даними. Він створює пряме посилання на більш ніж 40 джерел даних, розміщених в хмарі, таких як Amazon, Spark SQL, MySQL тощо. Tableau Reader – це безкоштовний інструмент, який дозволяє переглядати звіти і візуалізації, створені іншими компонентами. Дані можна фільтрувати, але редагування і модифікація обмежені.

Переваги та недоліки даної BI системи наведені у таблиці 2.3.

Таблиця 2.3 – Переваги та недоліки Tableau

Переваги	Недоліки
Інтуїтивно зрозумілий інтерфейс	Необхідність попередньої обробки даних, їх структуризації
Легкість інтеграції з багатьма сучасними Big Data платформами	Функції та опції системи вузько направлені, хоча повинні бути доступні широкому коли

(від Hadoop до Goodle BigQuery)	користувачів
Велика кількість вбудованих інструментів, які дозволяють імпортувати дані з різних джерел	
Підтримка на мобільних платформах	
Можливість спільної роботи над звітами	
Постійно оновлюється функціонал та покращуються існуючі характеристики, легкість установки оновлень	
Надійна служба підтримки клієнтів	Немає історії змін в Tableau Server
Велика кількість наявних відеоматеріалів та курсів	Необхідність в IT-консультуванні

2.3 Система бізнес аналізу Power BI

Power BI – це хмарна платформа для звітності і аналітики, інструмент для обробки даних з різних джерел, що забезпечує візуалізацію після процесу очищення та інтеграції. Неважливо, представлені дані простою таблицею Excel або колекцією хмарних і локальних гібридних сховищ даних, він дозволяє легко підключатися до джерел даних, візуалізувати важливі аспекти і надати доступ до результатів необхідним користувачам. Power BI позбавляє необхідності турбуватися про оновлення даних і версій файлів, дозволяючи в будь-який час отримувати доступ до звітів за допомогою комп'ютера, планшету або мобільного пристрою, адже складається з трьох компонентів: Desktop, Mobile-версії та веб-служби SaaS (рисунок 2.4) [10].

Рисунок 2.4 – Компоненти системи Power BI

Аналіз у Power BI базується на трьох інструментах: Power Query, PowerPivot та Power View.

Power Query – це технологія підключення до даних, яка дозволяє виявляти, підключати, комбінувати і уточнювати джерела даних для задоволення потреб в аналізі. PowerPivot – інструмент моделювання даних, який дозволяє створювати моделі, встановлювати зв'язки і здійснювати обчислення. За допомогою PowerPivot можна працювати з великими наборами даних, вибудовувати широкі зв'язки і створювати складні (або прості) обчислення в високопродуктивному середовищі і в звичних для Excel умовах. Power View – компонент візуалізації даних, який дозволяє створювати інтерактивні діаграми, графіки, карти та інші елементи. Він доступний в Excel, SQL Server, SharePoint та Power BI.

Основні можливості Power BI:

– збір інформації абсолютно з будь-яких джерел даних: сервіси, бази даних, файли, Google Docs, Excel, CSV, дані з Інтернету, API і різні інші коннектори;

– обробка даних, їх стандартизація;

– створення формул, параметрів та оцінок для аналізу ефективності діяльності підприємства та управління;

– інтерактивне відображення параметрів та оцінок для швидкого відслідковування та приймання заходів щодо їх покращення;

– можливість публікації всіх звітів і дашборда в Інтернеті за допомогою онлайн служби Power BI Service або через мобільний додаток;

– надання роздільних прав доступу для співробітників;

– використання серверних потужностей хмари Microsoft для автоматичної обробки будь-якої кількості даних;

– автоматичне оновлення всієї інформації (в моделі даних звітів), розміщеної в хмарі, що дозволяє отримувати актуальні дані в звітах Power BI в режимі онлайн;

– автоматичне сповіщення системою потрібних співробітників при досягненні критичних значень в заданих KPI.

Переваги та недоліки даної BI системи наведені у таблиці 2.4.

Таблиця 2.4 – Переваги та недоліки Power BI

Переваги	Недоліки
Доступність, наявна безкоштовна версія	При необхідності виконання операцій відмінних від базових, виникають труднощі
Інтеграція з іншими продуктами від Microsoft	Проблеми з імпортом і обробкою великих об’ємів даних, програма припиняє роботу або значно уповільнюється
Велика кількість вбудованих бібліотек візуалізації	
Можливість підключення майже будь-якого джерела даних (хмарного або офлайн)	
Інтуїтивний інтерфейс	

Загальні порівняльні оцінки (по 5-бальній шкалі) в результаті тестової експлуатації вищерозглянутих систем наведено у таблиці 2.5.

Таблиця 2.5 – Загальні порівняльні оцінки в результаті тестової експлуатації систем QlikView, Tableau, Power BI

Критерій	QlikView	Tableau	Power BI
Оцінка техпідтримки	5	5	0
Оцінка масштабованості	3	4	4
Оцінка підтримки великих об’ємів даних	3	5	5
Оцінка клієнтського доступу	5	5	2
Оцінка інтерфейсу	4	4	3
Оцінка інтегрування	4	5	3
Оцінка візуалізації	4	4	4
Оцінка моделювання і роботи аналітика	2	5	4
Оцінка адміністрування	4	5	4
Оцінка середовища розробки	5	2	5
Оцінка підтримки OLAP	2	5	4
Загальні оцінка			
Загальна оцінка перспектив впровадження	3	4	3
Загальні оцінка складності системи	4	5	4

З проведеного аналізу існуючих на ринку рішень бізнес-аналітичних систем можна зробити висновок, що усі вони є складними у використанні через наявну кількість функцій, підтримку роботи з різнорідними джерелами даних та необхідність працювати з великим обсягом інформації. Розглянуті системи отримали середню оцінку перспективи впровадження та мають ряд спільних недоліків, пов’язаних із взаємодією з користувачем: необхідність IT-консультування, труднощі в роботі з системою без спеціалізованих технічних знань та неочевидний синтаксис.

3 СЦЕНАРІЇ ВИКОРИСТАННЯ СИСТЕМИ

Перед розробкою безпосередньо бізнес-логіки програмного забезпечення необхідно визначити, що саме вміє робити система та які функціональні можливості вона надає користувачу. Діаграму сценаріїв використання системи аналізу бізнес-даних із використанням технологій обробки природних мов наведено у додатку А.

Передбачається, що взаємодіяти із системою аналізу бізнес-даних будуть наступні два актора:

– інженер даних – спеціаліст, обов’язками якого є налаштування та підтримка інфраструктури роботи з даними для подальшого їх використання бізнес-аналітиками;

– бізнес аналітик – спеціаліст, що використовує методи аналізу бізнес-даних для дослідження діяльності підприємства з метою визначення проблем та впровадження рішень на основі проведеного аналізу.

Систему аналізу бізнес-даних спроектовано таким чином, щоб максимально спростити взаємодію бізнес-аналітиків із програмним застосунком завдяки використанню технологій обробки природніх мов. Але для того, щоб мати змогу отримати необхідні дані, ці данні повинні бути попередньо завантажені у систему. Для цього потребується налаштування профілів з’єднання зі сторонніми джерелами даних, написання SQL-запитів тощо. Очевидно, що це потребує певної технічної кваліфікації і не кожен бізнес-аналітик матиме такі навички. Саме тому вводиться поняття інженера даних системи.

Розглянемо окремо кожен зі сценаріїв використання системи (прецедент). Першим сценарієм використання є авторизація інженера даних у системі. Детальний опис прецеденту наведено у таблиці 3.1.

Таблиця 3.1 – Опис прецеденту «Авторизація»

Назва	Авторизація
Опис	Надає можливість інженеру даних авторизуватись у системі
Актори	Інженер даних

Передумови	Інженер даних авторизований у системі
Головний сценарій	Інженер даних вводить логін та пароль. Після введення коректних даних, система авторизує інженера даних.
Виключний сценарій	Якщо введені дані не коректні, система повідомляє про це інженера даних.

Наступним сценарієм використання системи аналізу бізнес-даних є налаштування сховища або джерела даних (таблиця 3.2). На основі збережених даних при використанні цього сценарію, інженер даних має можливість наповнювати систему даними, з якими надалі будуть працювати бізнес-аналітики.

Таблиця 3.2 – Опис прецеденту «Налаштування сховища або джерела даних»

Назва	Налаштування сховища або джерела даних
Опис	Збереження, редагування або видалення інформації у системі про сховище даних (база даних із датасетами) або джерело даних (datasource)
Актори	Інженер даних
Передумови	Інженер даних авторизований у системі
Головний сценарій	1. При додаванні/редагуванні адміністратор заповнює інформацію, необхідну для підключення до сховища або джерела даних (URL сервера БД, ім’я користувача, пароль) та вказує назву, що ідентифікує об’єкт. При збереженні інформація записується у БД. 2. При видаленні інженер даних обирає відповідний об’єкт та натискає кнопку «видалити». Відповідний запис видаляється з бази даних.
Виключний сценарій	Якщо виникає помилка при модифікації даних, система повідомляє про це інженера даних.

Описаний вище сценарій використання розширюється прецедентом, який дозволяє перевірити можливість підключитися до зареєстрованого у системи сховища або джерела даних (таблиця 3.3). До того ж цей сценарій є самостійним (тобто може використовуватися незалежно від першого), оскільки сховища, джерела даних та система є компонентами у мережі, які взаємодіють між собою за допомогою технології JDBC, та адміністратор повинен мати можливість протестувати з’єднання між даними компонентами і у разі невдачі отримати інформативну помилку.

Таблиця 3.3 – Опис прецеденту «Перевірка підключення до сховища або джерела даних»

Назва	Перевірка підключення до сховища або джерела даних
Опис	Перевіряється можливість встановлення підключення у мережі між системою та зовнішньою базою даних
Актори	Інженер даних
Передумови	Інженер даних авторизований у системі
Головний сценарій	Інженер даних обирає потрібний профіль підключення до джерела або сховища даних та натискає кнопку «перевірити з’єднання». Система намагається встановити зв’язок із БД та повертає відповідний результат.

Наступним сценарієм використання є налаштування датасету (таблиця 3.4), шо включає у себе модифікацію інформації про даний об’єкт, а саме вибір джерела та сховища даних, визначення SQL-запиту на отримання даних із джерела.

Таблиця 3.4 – Опис прецеденту «Налаштування датасету»

Назва	Налаштування датасету
Опис	Збереження, редагування або видалення інформації у системі про датасет (назва, джерело даних, сховище даних та SQL-

	запит)
Актори	Інженер даних
Передумови	Інженер даних авторизований у системі
Головний сценарій	Інженер даних у формі датасету обирає джерело, з якого будуть завантажуватися дані, та сховище – база призначення датасету. Також заповнюється додаткова інформація.

Прецедент налаштування датасету розширюється сценарієм завантаження даних у датасет (таблиця 3.5), який у свою чергу включає у себе перевірку підключення як до джерела даних, так і до сховища.

Таблиця 3.5 – Опис прецеденту «Завантаження даних»

Назва	Завантаження даних
Опис	Завантаження даних у систему із джерела даних
Актори	Інженер даних
Передумови	Інженер даних авторизований у системі
Головний сценарій	У формі датасету при натисканні кнопки «Оновити дані» відбувається вилучення даних із джерела, та запис їх запис у відповідне сховище.
Виключний сценарій	Якщо при зборі або завантажені даних виникає помилка, система сповіщає про це інженера даних.

Наступним прецедентом системи є отримання даних зі сховища (таблиця 3.6), Даний сценарій підтримується у системі для того, щоб інженер даних міг переконатися, які саме дані вже знаходяться у сховищі.

Таблиця 3.6 – Опис прецеденту «Отримання даних зі сховища»

Назва	Отримання даних зі сховища
Опис	Отримання набору даних з існуючого датасету у сховищі

Актори	Інженер даних
Передумови	Інженер даних авторизований у системі
Головний сценарій	Інженер даних обирає об’єкт датасету, з якого необхідно отримати дані. При успішному результаті, інженер даних побачить таблицю із відповідним набором даних.
Виключний сценарій	Якщо при зборі даних виникає помилка, система сповіщає про це інженера даних.

Прецедент отримання даних включає прецедент перевірки підключення, оскільки необхідно переконатися, чи взагалі можливо підключитися до БД.

Останнім і ключовим прецедентом системи є отримання даних за запитом природньою мовою (таблиця 3.7). У цьому сценарії з’являється актор бізнес-аналітик.

Таблиця 3.7 – Опис прецеденту «Отримання даних за запитом природньою мовою»

Назва	Отримання даних за запитом природньою мовою
Опис	Отримання даних за допомогою запиту природньою мовою
Актори	Інженер даних, бізнес-аналітик
Передумови	Інженер даних або бізнес-аналітик має мессенджер із встановленим чат-ботом
Головний сценарій	Інженер даних або бізнес-аналітик вводить запит у мессенджері. Система інтерпретує цей запит у SQL за допомогою інструментів NLP та формує відповідь.
Виключний сценарій	Якщо система не «зрозуміла» запит або ж не знайшла відповідні об’єкти, то формується повідомлення про помилку.

4 СТРУКТУРНА СХЕМА СИСТЕМИ

Структурну схему системи аналізу бізнес-даних з використанням технологій оброблення природних мов наведено у додатку Б. Центральним об’єктом системи є сервер (рисунок 4.1), який складається з наступних чотирьох компонентів:

– модуль роботи із джерелами даних;

– модуль роботи зі сховищами даних;

– модуль представлення даних (чатбот);

– внутрішня база даних.

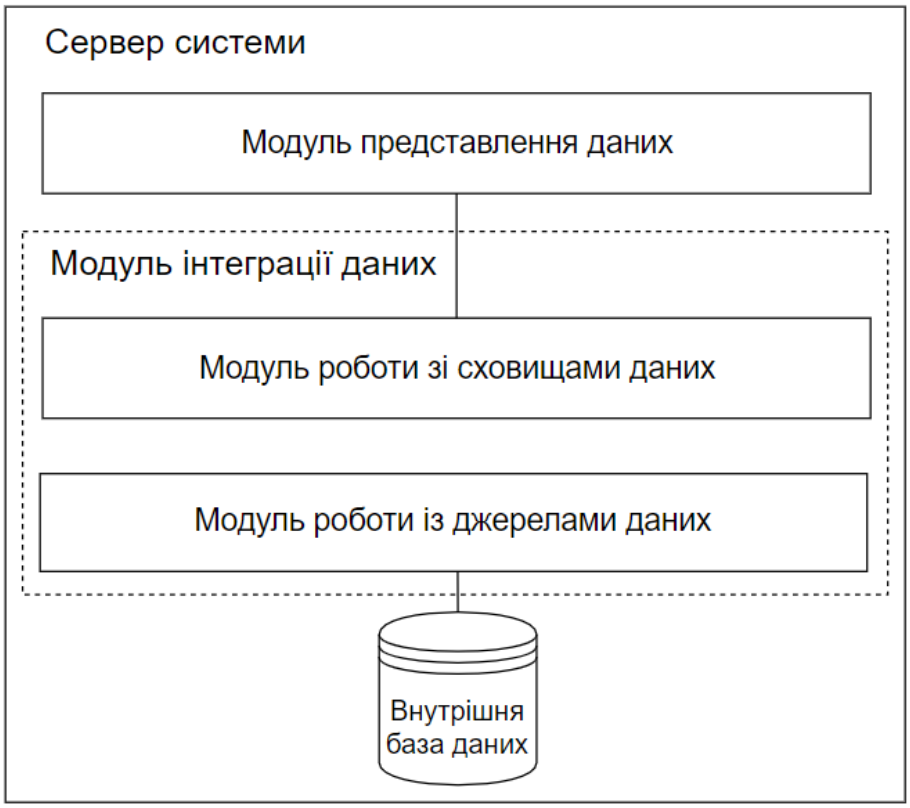

Рисунок 4.1 – Структура серверу системи аналізу бізнес-даних

Перші два модулі доцільно угрупувати в один логічний об’єкт – модуль інтеграції даних, який є невід’ємною частиною системи, оскільки виникає необхідність збереження структурованих бізнес-даних у одному місці, завдяки чому модуль представлення даних матиме єдиний уніфікований інтерфейс доступу до них

в залежності від обраної бази даних. Джерелами даних можуть виступати бази даних, які підтримують JDBC інтерфейс доступу до даних.

На першому етапі дані витягуються з обраної підтримуваної бази даних і готуються до збереження. Слід зазначити, що для коректного представлення даних після їх завантаження із бази у систему повинні вилучатись не тільки самі дані, але й інформація, що описує їх структуру, з якої будуть сформовані метадані їх збереження.

У якості типів даних було обрано найбільш популярні, які підтримуються більшістю сховищами даних:

– цілочисельні;

– десяткові;

– текстові;

– дати.

Наступним етапом інтеграції даних є збереження даних у сховище. Сховищами даних також виступають реляційні бази даних. Розроблену систему спроектовано таким чином, що окрім стандартного сховища, яке постачається із системою за замовчуванням, реалізовано можливість підключення зовнішніх сховищ даних, які було б зручно використовувати клієнтам системи в залежності від інфраструктури на підприємстві.

Для інтеграції даних у системі реалізовано REST API, за допомогою якого інженер даних матиме можливість завантажувати бізнес-дані у систему.

Представлення даних відбувається за допомогою корпоративного месенджера Slack. Бізнес-аналітик надсилає повідомлення боту, яке спочатку потрапляє на сервер Slack, а потім перенаправляється на встановлений додаток на сервер. Далі виконуються алгоритм обробки запиту. По-перше, відбувається парсинг користувацького запиту за допомогою інструментів обробки природних мов. Наступним кроком встановлюється співвідношення між запитом та існуючими у системі даними про датасет. На основі цього генерується та виконується запит на отримання набору даних. Дані приводяться до потрібного формату та повертаються користувачу у вигляді повідомлення.

5 ВИБІР ТА ОБГРУНТУВАННЯ ЕЛЕМЕНТІВ ТА ТЕХНОЛОГІЙ

Основними критеріями вибору мови програмування для розробки системи управління сховищами даних із використанням технологій обробки природних мов є:

- платформа для розробки серверного програмного забезпечення;
- наявність інструментів обробки природних мов.

Мова програмування Java відповідає усім зазначеним вище потребам. Spring-фреймворк забезпечує легку розробку серверного застосунку та надає широкий спектр можливостей у кастомізації додатку. Також існує велика кількість бібліотек, які надають інструменти обробки природних мов. Серед усіх було обрано бібліотеку Stanford NLP.

Stanford NLP надає набір інструментів аналізу природної мови, написаних на Java. Він може приймати необроблений текст людською мовою та давати основні форми слів, їх частини мови, будь то назви компаній, людей тощо, нормалізувати та інтерпретувати дати, час та числові величини, розмічати структуру речень з точки зору фраз або залежностей від слів, і вкажіть, які іменні фрази стосуються тих самих сутностей. Спочатку він був розроблений для англійської мови, але зараз також забезпечує різний рівень підтримки арабської, китайської, французької, німецької та іспанської мов. Stanford NLP – це інтегрований фреймворк, завдяки якому дуже легко застосовувати купу інструментів мовного аналізу до фрагмента тексту. Також це набір стабільних і перевірених засобів обробки природних мов, що широко використовуються різними групами в наукових колах, промисловості та уряді. Інструменти по-різному використовують засновані на правилах, імовірнісне машинне навчання та компоненти глибокого навчання [11].

Описана вище бібліотека надає наступні функціональні можливості:

- токенізація – процес перетворення тексту на лексеми;
- розбиття речення – процес поділу тексту на речення;

– позначення частини мови (POS-tagging) – присвоює лексемам мітку частини мови, наприклад, дієслово чи іменник;

– лематизація – відображає словникову форму слова (лему);

– розпізнавання іменованих сутностей – розпізнає в тексті іменовані сутності (назви осіб, компаній тощо). В основному цей анотатор використовує одну або кілька моделей послідовності машинного навчання для позначення сутностей, але він може також викликати спеціалізовані компоненти, засновані на правилах, наприклад, для маркування та інтерпретації часу та дат;

– розбір залежностей – забезпечує швидкий синтаксичний аналіз залежностей;

– відкрите вилучення інформації (openEI) – витягує потрійні відношення: що представляють предмет, відношення та об’єкт відношення.

Основною можливістю бібліотеки, яка використовувалась при розробці алгоритму парсингу запиту природною мовою, є POS-tagging. У таблиці 5.1 наведені можливі мітки частин мови та приклади [12].

Таблиця 5.1 – Мітки POS-tagging

Тег	Опис	Приклад	Тег	Опис	Приклад
CC	coordin.conjuction	and, but, or	SYM	symbol	+, %, &
CD	cardinal number	one, two	TO	«to»	to
DT	determiner	a, the	UH	interjection	ah, oops
EX	existential «there»	there	VB	verb base from	eat
FW	foreign word	mea cupla	VBD	verb past tense	ate
IN	preposition/sub-conj	of, in, by	VBG	verb gerund	eating
JJ	adjective	yellow	VBN	verb past participle	eaten
JJR	adj., comparative	bigger	VBP	verb non-3sg pres	eat
JJS	adj., superlative	wildest	VBZ	verb -3sg pres	eats
LS	list item marker	1, 2, One	WDT	wh-determiner	which, that
MD	modal	can, should	WP	wh- pronoun	what, who

NN	noun, sing. or mass	llama	WP$	possessive wh-	whose
NNS	noun, plural	llamas	WRB	wh- adverb	how, where
NNP	proper noun, sing.	IBM	$	dollar sign	$
NNPS	proper noun, plural	Carolinas	#	pound sign	#
PDT	predeterminer	all, both	“	left quote	‘ or “
POS	possessive ending	`s	”	right quote	’ or ”
PRP	personal pronoun	I, you, he	(	left parenthesis	[, {, (, <
PRP$	possessive pronoun	your, one`s	)	right parenthesis	], },), >
RB	adverb	quickly, never	,	comma	,
RBR	adverb, comparative	faster	.	sentence-final punc	. ! ?
RBS	adverb, superlative	fastest	:	mid-sentence punc	: ; … – -
RP	particle	up, off			

Ще одним важливим аспектом у розробці системи аналізу бізнес-даних з використанням технологій обробки природних мов є вибір системи управління базою даних, адже дані – основний операнд у бізнес-аналітиці. У більшості сучасних додатків використовують декілька систем управління базами даних, щоб скомбінувати та задіяти їх сильні сторони, адже немає одної, яка була б оптимізована для всіх можливих сценаріїв використання. Серед існуючих СУБД найбільш популярними є MySQL та PostgreSQL (рисунок 5.1) [13].

MySQL користується популярністю, тому що забезпечує високу швидкість обробки інформації, функціональність СУБД та масштабованість. До того ж, вона являє собою програмне забезпечення з відкритим кодом і тому досить гнучка (адже можна вдосконалювати код, вносити власні корективи) та доступна будь-кому безкоштовно. Для системи аналізу бізнес-даних з використанням технологій обробки природних мов MySQL є ідеальним варіантом ще й тому, що вона легко

інтегрується з фреймворком Spring та взаємодіє без проблем інтерфейсом JDBC API [14].

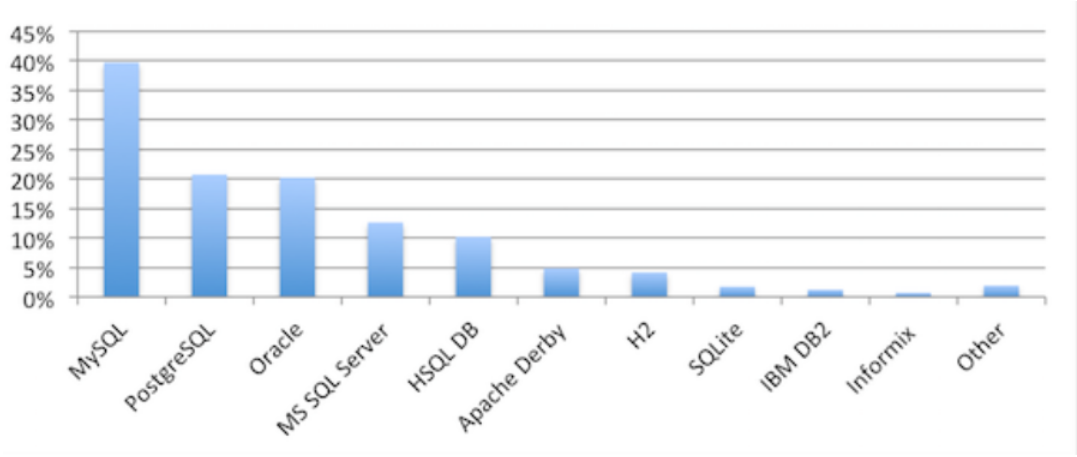

Рисунок 5.1 – Найбільш популярні СУБД

PostgreSQL – ще одна популярна реляційна СУБД, яка в першу чергу орієнтована на сумісність зі стандартами та масштабованість. Особливості цієї системи – це об’єктно-орієнтований функціонал та легкість обробки одночасно декількох задач. До того ж PostgreSQL забезпечує підтримку бази даних необмеженого розміру: немає обмежень на максимальний розмір БД, кількість записів та індексів у таблиці, що для сфери бізнес-аналітики є значною перевагою [15].

У якості додатку для обміну повідомленнями обрано Slack не тільки тому що він підтримує розробку чатботів, але й тому що він є корпоративним месенджером, і за допомогою воркспейсів Slack є можливість забезпечення ізольованості при обміні даними, оскільки для доступу до воркспейсу необхідно отримати пряме запрошення. Таким чином, на рівні Slack системою передбачено забезпечення конфіденційності даних між різними відділами одного підприємства.

6 ER-ДІАГРАМА

Основною можливістю системи аналізу бізнес-даних є створення датасетів, дані для яких отримуються із зовнішніх джерел та зберігаються у БД. Ці дані формуються динамічно та зберігаються у вигляді окремих таблиць. Таким чином проектування сутностей та визначення стовпців датасетів повністю покладається на інженера даних.

Але окрім власне даних датасету, є необхідність збереження метаданих у системі, які будуть використовуватися при обробці користувацького запиту природною мовою (назва датасету, назви колонок тощо). Також є необхідність збереження інформації, необхідної для підключення до зовнішніх баз даних, які будуть використовуватися адміністратором при налаштуванні датасетів. З цього випливає необхідність у статичній базі даних, з якою взаємодіятиме система аналізу бізнес-даних.

Визначимо окремо кожну з сутностей та відношення між ними. Перші дві сутності – це «Джерело даних» (таблиця 6.1) та «Сховище» (таблиця 6.2). Обидві інкапсулюють усю необхідну інформацію для підключення до БД на основі JDBC API. Загальну інформацію, що стосується JDBC драйвера, виділено у окрему сутність «JDBC драйвер» (таблиця 6.3). Відношення між джерелом даних або сховищем та JDBC драйвером «один до багатьох» (рисунок 6.1).

Таблиця 6.1 – Таблиця «Джерело даних»

Поле	Опис	Тип даних	Первинний ключ	Зовнішній ключ
jdbc_data_source_id	Ідентифікатор джерела даних	Integer	X	
name	Назва джерела даних	Varchar		
username	Ім’я користувача	Varchar		

password	Пароль користувача	Varchar		
url	URL підключення до БД за допомогою JDBC	Varchar		
jdbc_driver_id	Ідентифікатор JDBC драйвера	Integer		X

Таблиця 6.2 – Таблиця «Сховище»

Поле	Опис	Тип даних	Первинний ключ	Зовнішній ключ
storage_id	Ідентифікатор сховища	Integer	X	
name	Назва сховища датасетів	Varchar		
storage_type	Тип сховища (PostgreSQL або MySQL)	Varchar		
username	Ім’я користувача	Varchar		
password	Пароль користувача	Varchar		
url	URL підключення до БД за допомогою JDBC	Varchar		
jdbc_driver_id	Ідентифікатор JDBC драйвера	Integer		X

Таблиця 6.3 – Таблиця «JDBC драйвер»

Поле	Опис	Тип даних	Первинний ключ	Зовнішній ключ
jdbc_driver_id	Ідентифікатор JDBC драйвера	Integer	X	
class_name	Назва класу JDBC драйвера	Varchar		
current_time_query	SQL-запит для отримання поточного часу (використовується	Text		

	для перевірки підключення)			
name	Назва JDBC драйвера	Varchar		

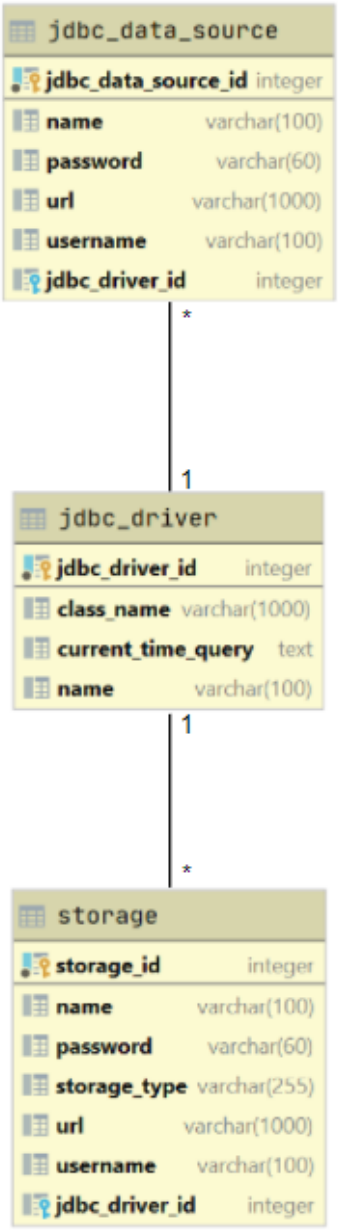

Рисунок 6.1 – Відношення між таблицями «Джерело даних», «Сховище» та «JDBC драйвер»

Важливою відмінністю між сутностями «Джерело даних» та «Сховище» є те, що остання має поле «тип сховища». При роботі із джерелом виконується лише зчитування даних, і адміністратор має можливість задавати SQL-запит для їх вилучення. Цей процес є загальним для усіх баз даних, які підтримують JDBC API, і

не потребується окремо інформація, до якого саме типу бази даних виконується запит. Єдиним обмеженням є те, які саме JDBC драйвери встановлено у системі аналізу бізнес-даних. На відміну від цього робота зі сховищем даних потребує не тільки читання даних, але й запис датасетів у базу. Більш того, для зчитування даних з датасетів використовуються запити природною мовою, що потребують їхньої конвертації у відповідні SQL-запити. Діалекти цих запитів можуть суттєво різнитися в залежності від БД, тому підтримка різних сховищ потребує певного розширення коду для його підтримки у системі. Саме тому вводиться поняття типу сховища для відповідної сутності. У рамках даної роботи система аналізу бізнес даних підтримує два типи сховища даних – PostgreSQL та MySQL.

Наступними сутностями є «Датасет» та «Колонка датасета», які містять інформацію про набір даних, які використовуються при пошуку потрібного датасету підчас виконання алгоритму парсингу запиту. Ці дані заздалегідь налаштовуються адміністратором системи. Відношення між сутностями – «один до багатьох» (рисунок 6.2). Детальний опис сутностей наведено у таблицях 6.4 та 6.5.

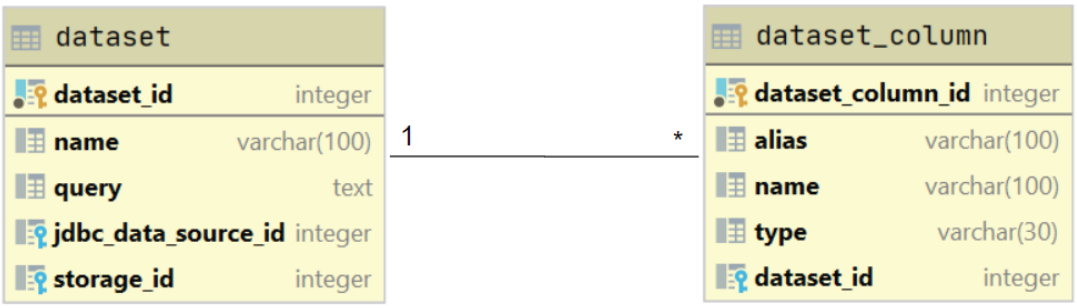

Рисунок 6.2 – Відношення між таблицями «Датасет» та «Колонка датасета»

Таблиця 6.4 – Таблиця «Датасет»

Поле	Опис	Тип даних	Первинний ключ	Зовнішній ключ
dataset_id	Ідентифікатор датасета	Integer	X	

name	Назва датасета	Varchar		
query	SQL-запит для отримання даних із джерела	Text		
storage_id	Ідентифікатор сховища	Integer		X
jdbc_data_source_id	Ідентифікатор джерела даних	Integer		X

Таблиця 6.5 – таблиця «Колонка датасета»

Поле	Опис	Тип даних	Первинний ключ	Зовнішній ключ
dataset_column_id	Ідентифікатор колонки датасета	Integer	X	
name	Назва колонки	Varchar		
type	Тип даних колонки	Varchar		
alias	Псевдонім колонки	Varchar		
jdbc_data_source_id	Ідентифікатор джерела даних	Integer		X

Повну ER-діаграму наведено у додатку В.

7 РЕАЛІЗАЦІЯ БІЗНЕС-ЛОГІКИ СИСТЕМИ

7.1 Модуль роботи із джерелами даних

Повну діаграму класів модулю роботи із джерелами даних наведено у додатку Г. Даний модуль розроблено таким чином, щоб при необхідності у систему аналізу бізнес-даних було можливо легко інтегрувати інші джерела даних. Досягнуто це за допомогою поліморфізму – як одного з базових принципів ООП, та за допомогою Spring-фреймворку, а саме завдяки технології впровадження залежностей (Dependency Injection). Архітектуру класів клієнтів наведено на рисунку 7.1.

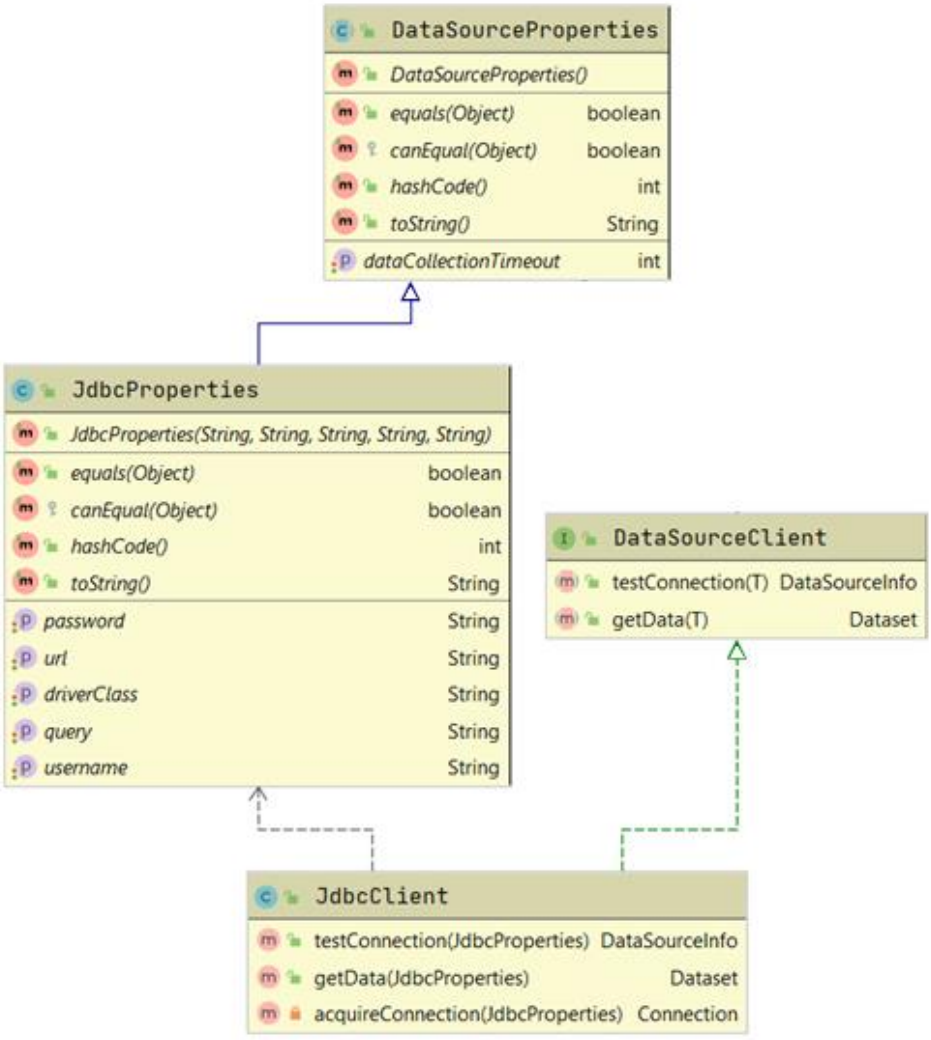

Рисунок 7.1 – Діаграма класів клієнтів джерел даних

Інтерфейс DataSourceClient описує загальну поведінку усіх джерел даних та містить два методи:

– testConnection;

– getData.

Перший метод потрібен для того, щоб адміністратор системи міг переконатися, чи є введені ним дані для підключення до джерела даних валідними та чи є можливість у системи підключитися до нього. У якості результату проведення перевірки з'єднання цей метод повертає об'єкт класу DataSourceInfo. Для кожної імплементації клієнта є можливість розширення цього класу шляхом додавання специфічної інформації конкретного джерела даних. На рисунку 7.2 наведено приклад для JDBC клієнта:

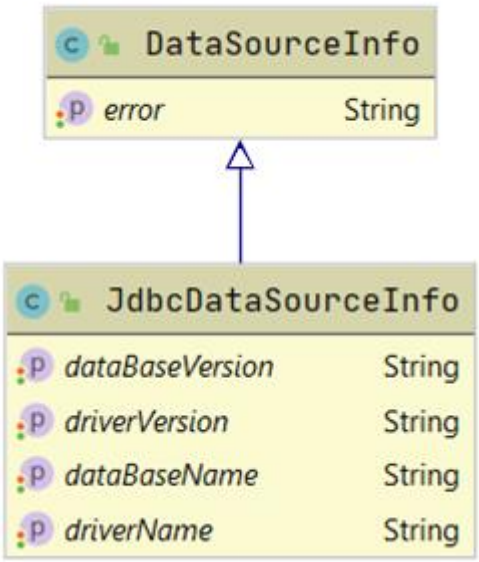

Рисунок 7.2 – Діаграма класів інформації про джерело даних

Загальним полем для всіх джерел даних є поле error, яке є незаповненим у разі вдалого встановлення з'єднання, або ж містить інформацію про помилку, яка виникла при зворотньому випадку. Специфічною інформацією для джерела даних JDBC є:

– dataBaseVersion – версія БД;

– driverVersion – версія JDBC драйвера;

– dataBaseName – назва БД;

– driverName – назва JDBC драйвера.

Усі ці поля отримуються з об’єкту DatabaseMetaData, який у свою чергу отримується з об’єкту Connection. Слід зазначити, що DatabaseMetaData та Connection – це частина Java API, що описує роботу із базами даних (переважно реляційними), і кожен JDBC драйвер імплементує ці інтерфейси для кожної БД окремо.

На рисунку 7.3 наведено діаграму класів, що демонструє результат роботи методу getData.

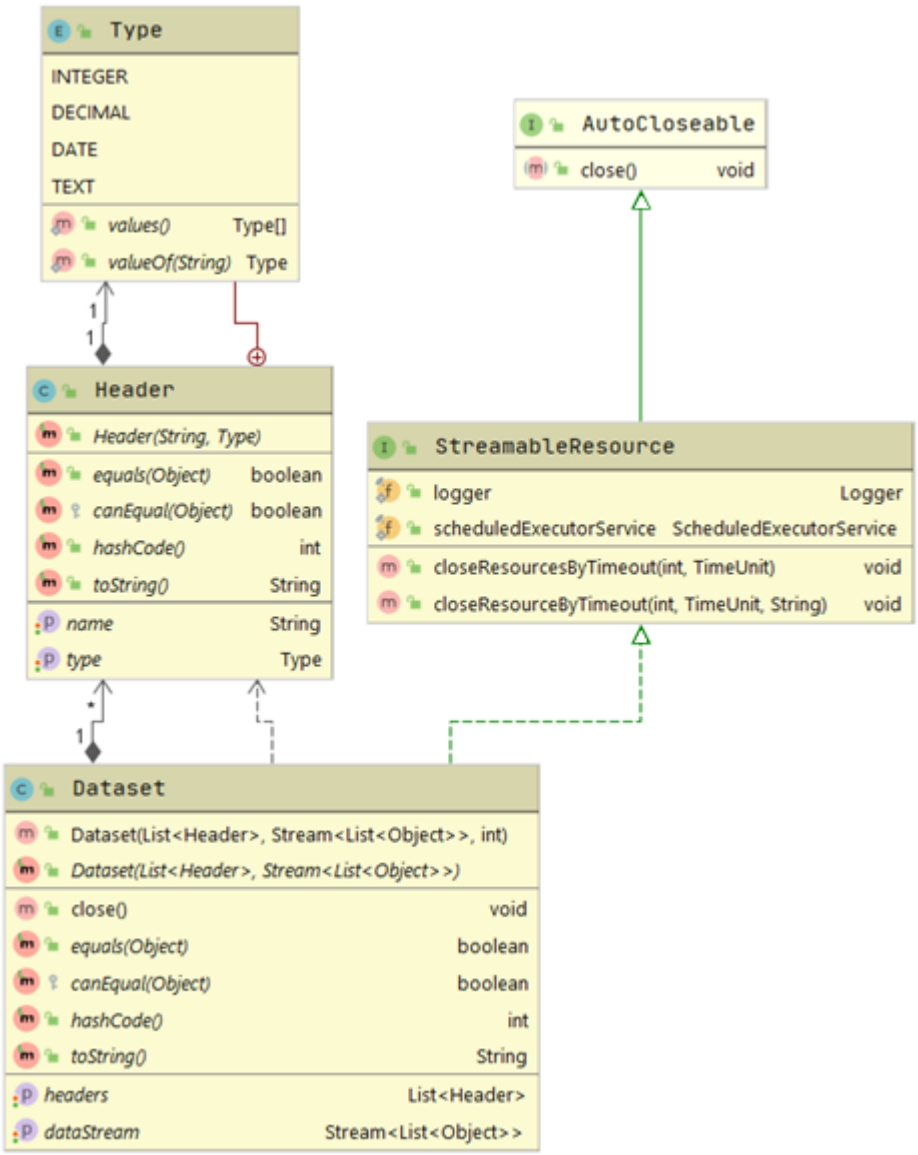

Рисунок 7.3 – Діаграма класів набору даних

Другий метод інтерфейсу DataSourceClient – getData, виконує безпосередньо вилучення даних з джерела. На відміну від методу testConnection, результатом роботи методу getData є об’єкт класу Dataset, який є уніфікованим для всіх джерел даних і для кожної реалізації клієнта немає необхідності його розширення. Завдяки такому підходу спрощується подальша обробка цих даних, а саме збереження їх у сховищі та представлення цих даних користувачу.

Клас Dataset має два поля:

– headers;
– dataStream.

З точки зору реляційної моделі даних кожен об’єкт класу Dataset є відношенням, де поле headers – це список атрибутів, а поле dataStream – це набір кортежів.

Перше поле – це список об’єктів типу Header, який містить інформацію про ім’я колонки та її тип. Кожна реляційна база даних має різноманітну кількість типів даних, які відрізняються певними властивостями, але для всіх можна виділити основні типи даних:

– цілочисельні;
– десяткові;
– текстові;
– дати.

У програмній реалізації описані вище типи визначені за допомогою спеціального типу даних – перечислення (enum), який дозволяє змінній бути набором заздалегідь визначених констант. Також на діаграмі класів видно, що перечислення Type є внутрішнім типом класу Header.

Поле dataStream має тип Stream<List<Object>>, що можна трактувати як потік зі списку об’єктів. Типом цього поля обрано Stream, тому що це дозволяє динамічно отримувати дані з датасету без повного їх завантаження у пам’ять. Деякі імплементації Stream вимагають явного закриття ресурсів, що змушує розробника викликати метод close або поміщати об’єкт цього типу у конструкцію try-with-resources, яка автоматично викличе метод close. Для того, щоб не виникало ситуації,

коли за якихось обставин ресурси не були закриті (наприклад, виникнення помилкі до обробки потоку даних), створено інтерфейс StreamableResource. Він містить стандартний метод closeResourcesByTimeout, призначення якого примусово закривати ресурси через заданий проміжок часу. Досягається це за допомогою ScheduledExecutorService з пакету java.util.cuncurrent, який дозволяє сконфігурувати інтервал або час, на яке буде відкладено виконання задачі. На рисунку 7.4 наведено лістинг коду, який демонструє описаний вище метод.

```
default void closeResourcesByTimeout(final int timeout, final TimeUnit timeUnit) {
    scheduledExecutorService.schedule(() -> {
        try {
            logger.info("Closing stream resource after timeout: {} {}", timeout, timeUnit);
            this.close();
        } catch (Exception e) {
            logger.error("Couldn't close streamable resource: {}", e.getMessage());
            throw new DataSourceException("Couldn't close streamable resource: " + e.getMessage(), e);
        }
    }, timeout, timeUnit);
}
```

Рисунок 7.4 – Фрагмент коду метода closeResourcesByTimeout

Цей метод викликається у конструкторі класу Dataset, тобто одразу після створення об’єкту датасету починається відлік часу, після якого ресурси будуть закриті та буде неможливо вичитати дані з датасету.

Інтерфейс DataSourceClient є параметризованим. Узагальнення (generics) у мові програмування Java дозволяють типам або методам працювати із об’єктами різних типів, при цьому забезпечуються «безпека типів» на етапі компіляції. У реалізованому в системі інтерфейсі даний підхід використовується для узагальнення аргументів, визначених у методах. На рисунку 7.5 представлено, як це реалізовано синтаксично.

```
public interface DataSourceClient<T extends DataSourceProperties> {
    DataSourceInfo testConnection(T properties);
    Dataset getData(T properties);
}
```

Рисунок 7.5 – Фрагмент коду інтерфейса DataSourceClient

Такий підхід забов’язує розробника не тільки імплементувати методи інтерфейсу, а й наслідувати клас DataSourceProperties та визначати його у якості параметра при оголошенні класу клієнта. Конструкція «T extends DataSourceProperties» говорить про те, що параметром класу, який реалізує інтерфейс, може бути лише тип, який є нащадком класу DataSourceProperties, у інакшому випадку виникне помилка компіляції.

Ідея даного параметру полягає у тому, що для кожної імплементації клієнта джерела даних при виконанні будь-якого з методів, аргумент properties містить усю необхідну інформацію, що використовується для підключення до джерела даних та отримання датасетів. У випадку JDBC клієнта, створено сутність JdbcProperties, яка містить наступні поля:

– url;
– username;
– password;
– driverClass;
– query.

Перші чотири параметри необхідні для встановлення підключення із базою інструментами JDBC та створення об’єкту Connection. Останній параметр містить SQL-запит, за допомогою якого отримуються дані та формується об’єкт Dataset.

Базовий для всіх клієнтів клас DataSourceProperties містить одне єдине поле – dataCollectionTimeout, яке визначає таймаут на закриття потоку даних об'єкту Dataset, що було зазначено вище. Цей параметр має цілочисельний тип та завжди становить триста секунд (що дорівнює п'яти хвилинам), але при необхідності адміністратору можна надати можливість кастомізації даного параметру.

На рисунку 7.6 наведено діаграму класів, що демонструють принцип роботи методу getData у імплементації клієнта джерел даних JdbcClient. Алгоритм вилучення даних за допомогою інструментів JDBC:

а) встановлення підключення до БД. На даному етапі викликається приватний метод acquireConnection, який на основі необхідної для встановлення з’єднання

інформації з сутності JdbcProperties створює об’єкт Connection та повертає його у якості результату;

б) виконання SQL-запиту. На основі об’єкту Connection, отриманого на попередньому етапі, створюється об’єкт Statement, якому передається запит у вигляді строки. Результатом цього шагу є об’єкт ResultSet.

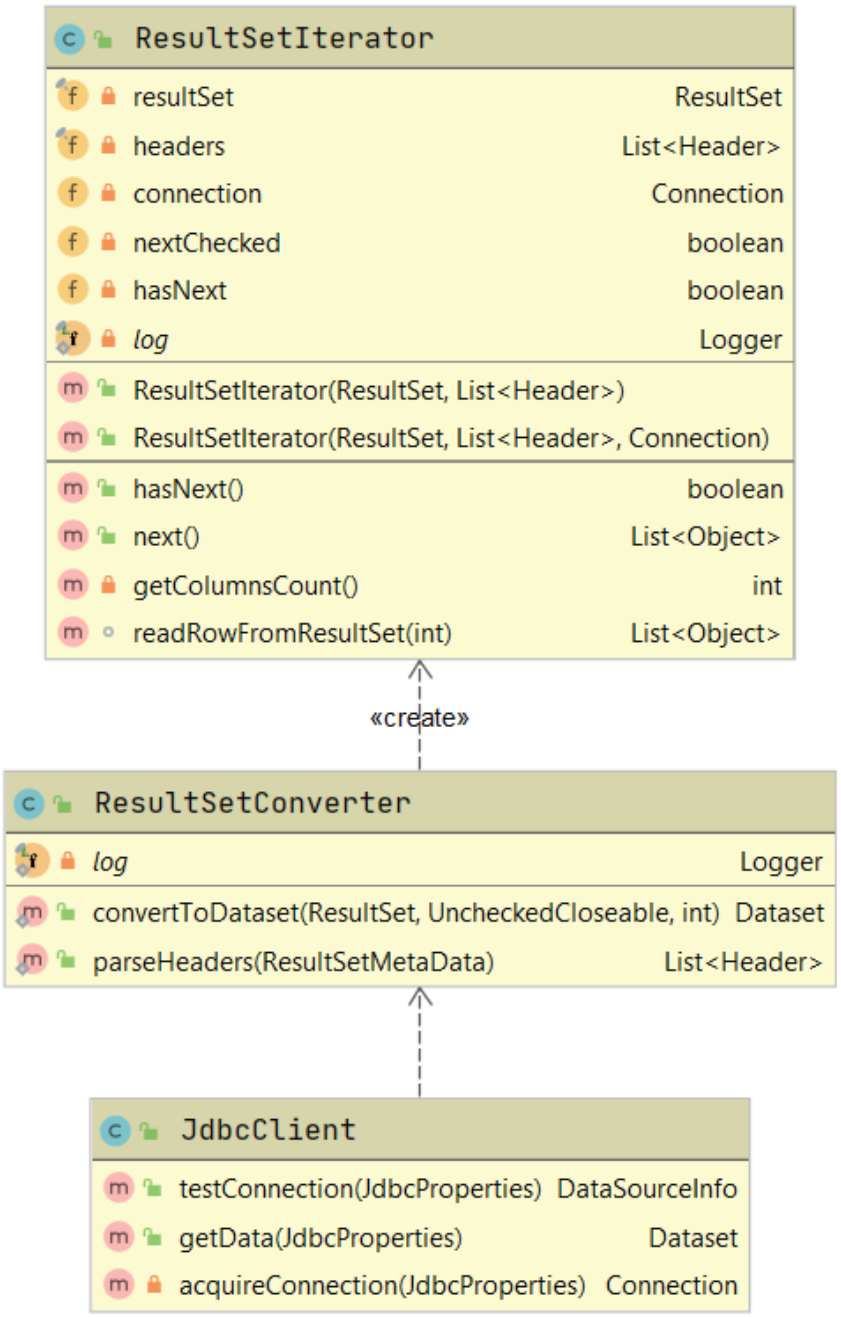

Рисунок 7.6 – Діаграма класів, що демонструє принцип роботи методу getData

в) створення об’єкту Dataset. Допоміжний клас ResultSetConverter на основі об’єкту ResultSet ініціалізує новий об’єкт Dataset.

Методи класу ResultSetConverter є статичними, тому ініціалізації об’єкту не виконується. Більш того, оскільки ResultSetConverter є Utility класом, для того щоб не можна було створювати екземпляри даного класу, він має приватний конструктор без аргументів. Зроблено це за допомогою анотації з бібліотеки Lombok, що дозволяє писати лаконічний код мовою Java (рисунок 7.7).

```
@NoArgsConstructor(access = AccessLevel.PRIVATE)
public class ResultSetConverter {
```

Рисунок 7.7 – Фрагмент коду класу ResultSetConverter

Першим кроком на етапі створення об’єкту Dataset, виконується ініціалізація поля headers. Виконується це за допомогою методу parseHeaders, який приймає у якості аргументу метадані з об’єкту ResultSet, та повертає список заголовків. З об’єкту метаданих отримується інформація про ім’я та тип колонки. У пакеті java.sql клас Types налічує понад тридцяти різних типів, які приводяться до основних чотирьох типів, які підтримуються розробленою системою аналізу бізнес-даних. Допоміжний метод convertJdbcColumnType виконує маппінг JDBC типів до чотирьох основних типів. Метод містить синтаксичну конструкцію switch expression, завдяки чому блок switch може повертати результат. За допомогою цього код виглядає більш лаконічним та зрозумілим (рисунок 7.8).

```
public static Header.Type convertJdbcColumnType(int columnType) {
    return switch (columnType) {
        case Types.INTEGER, Types.SMALLINT, Types.BIGINT, Types.TINYINT -> Header.Type.INTEGER;
        case Types.DECIMAL, Types.DOUBLE, Types.FLOAT, Types.NUMERIC, Types.REAL -> Header.Type.DECIMAL;
        case Types.DATE, Types.TIME, Types.TIMESTAMP -> Header.Type.DATE;
        default -> Header.Type.TEXT;
    };
}
```

Рисунок 7.8 – Фрагмент коду метода convertJdbcColumnType

Наступним кроком виконується створення потоку даних на основі об’єкту ResultSet. Для цього використано патерн поведінки – ітератор. Цей шаблон надає послідовний доступ до всіх елементів складеного об’єкта без розкриття його внутрішнього представлення [16]. Як само реалізовано цей патерн зображує діаграма класів з рисунку 7.9.

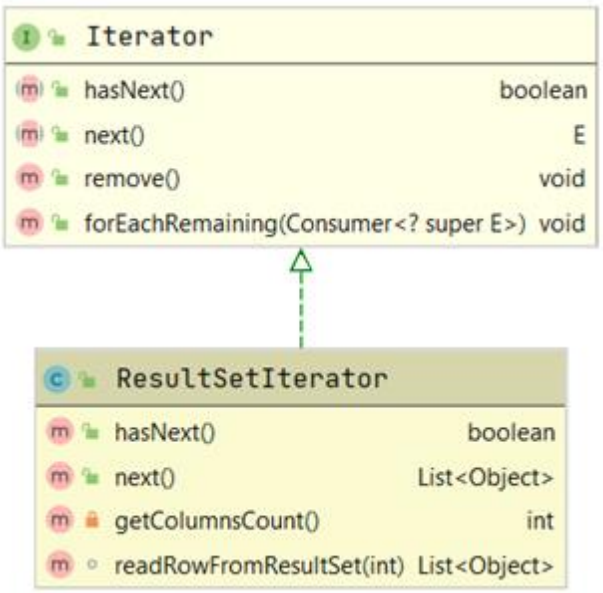

Рисунок 7.9 – Діаграма класів патерну ітератор

Інтерфейс Iterator є параметризованим, де E – це тип, який повертає ітератор при виклику методу next. У випадку класу ResultSetIterator цим типом виступає список об’єктів (List<Object>).

Об’єкт ResultSetIterator містить у собі об’єкт ResultSet, з якого виконується зчитування даних. Недоліком такого підходу є те, що поки ітератор не поверне усі елементи послідовності (тобто поки метод hasNext повертає значення true), об’єкт ResultSet не буде закрито, а це у свою чергу означає, що із БД буде встановлено з’єднання. Але даний підхід дозволяє уникнути обмеження розміру датасету за пам’ятью. При розробці системи аналізу бізнес-даних перевагу було віддано саме тому підходу, який не обмежує розмір даних за пам’яттю, оскільки більш важливим показником системи є те, який саме об’єм даних може бути опрацьованим при їх завантаженні у сховища даних.

Кожен клас-імплементація інтерфейсу клієнта джерела даних має містити анотацію Service. Це анотація Spring-фреймворку, за допомогою якої реалізується патерн фасад. Шаблон фасад – це структурний шаблон проектування, який дозволяє скрити складність системи завдяки зведенню усіх можливих зовнішніх викликів до одного об’єкту, що делегує їх відповідним об’єктам у системі []. Також дана анотація дозволяє використовувати впровадження залежностей. Завдяки цьому немає необхідності кожен раз створювати новий об’єкт, натомість Spring-фреймворк впровадить цей об’єкт, взявши готовий екземпляр з DI-контейнеру або створить новий, в залежності від того, який обрано цикл життя біна. Налаштовується це за допомогою анотації Scope. За замовчуванням – це singleton, тобто при роботі системи буде існувати тільки один екземпляр класу. На рисунку 7.10 наведено діаграму класів, на якій продемонстровано залежність між класами, де використовується впровадження залежностей, а саме клієнту джерела даних.

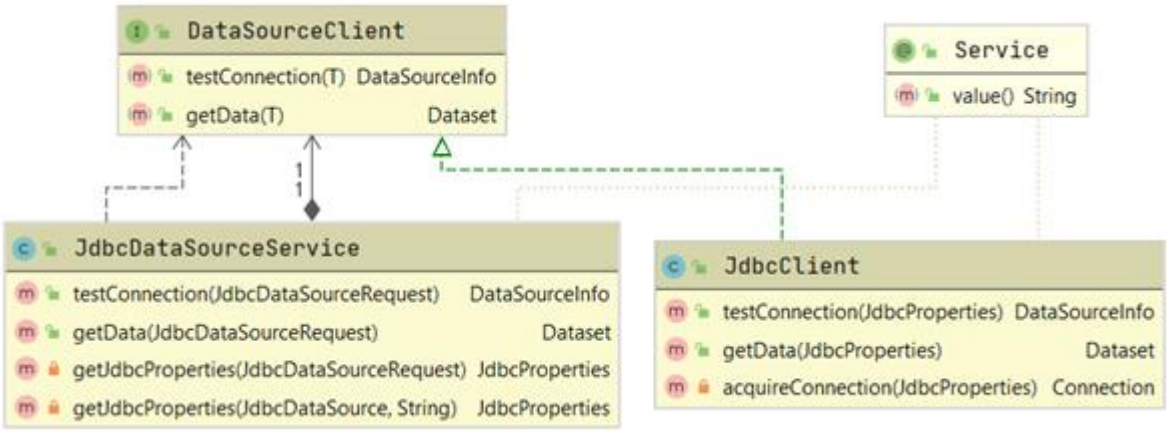

Рисунок 7.10 – Діаграма класів, що демонструє використання впровадження залежностей

Для клієнт-серверної взаємодії користувацького інтерфейсу з сервером створено контролер, який має два POST методи, що відповідають двом сценаріям використання – перевірка підключення до джерела даних та отримання даних з нього. Обидва методи у якості тіла запиту приймають об’єкт JdbcDataSourceRequest, який містить ідентифікатор, якому відповідає запис у БД із інформацією про джерело даних, та SQL-запит, якщо поточним методом є отримання даних.

Екземпляр класу JdbcDataSourceRequest є об’єктом передачі даних (DTO). Даний патерн використовується для передачі даних між модулями системи і на відміну від об’єкту бізнес-логіки, він не має поведінки.

Структура проекту, який розроблено за допомогою фреймворку Spring, є стандартною. Вона розбита не декілька рівнів, кожен з яких виконує власну функцію. У таблиці 7.1 наведено список рівнів, їх призначення, та приклади класів, які відповідають наведеним рівням у модулі роботи із джерелами даних.

Таблиця 7.1 – Стандартні рівні Spring-проекту

Рівень	Призначення	Приклад класів
Model	З архітектурного патерну MVC, рівень model представляє дані – сутності бізнес-логіки системи. На цьому рівні використовується така технологія, як об’єктно-реляційне відображення (ORM), що пов’язує сутності БД із класами.	JdbcDataSource, JdbcDriverMetadata
Repository	Забезпечує спрощений доступ до даних, що представляють собою концептуальну множину. Його поведінка схожа на поведінку колекції, за винятком більш розвинених можливостей для побудови запитів [16].	JdbcDataSourceRepository
Service	Рівень сервісів інкапсулює певну поведінку бізнес-логіки та реалізує патерн фасад.	JdbcDataSourceService
Controller	З архітектурного патерну MVC, рівень контролера реагує на дії користувача та оповіщає модель про необхідність зміни [17]. У випадку Spring-фреймворку контролер приймає HTTP запити, та за допомогою анотацій вказується, який саме метод та URI використовуються.	DataSourceController

На рисунку 7.11 наведено діаграму, що показує зв’язок між класами – компонентами кожного з рівня. При виконанні HTTP запиту, контролер на основі отриманої інформації передає запит відповідному сервісу, який виконує певну логіку. У даному випадку, сервіс отримує дані з БД, звертаючись до репозиторію, та на основі отриманого результату генерує запит до джерела даних та повертає результат контролеру, який у свою чергу повертає відповідь у форматі JSON. У випадку, якщо за заданим ідентифікатором запису у БД не знайдено – генерується виключення DataSourceException, яке розширює базове виключення зі стандартного пакету java RuntimeException.

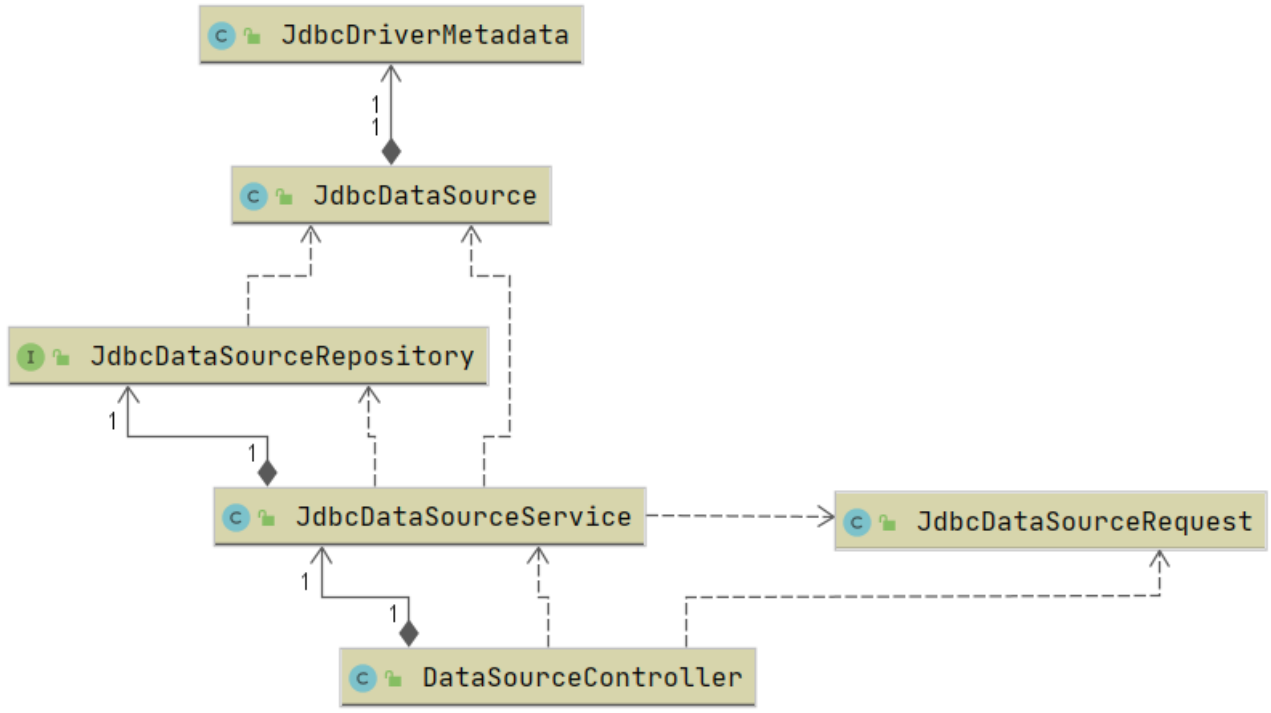

Рисунок 7.11 – Діаграма класів, що демонструють зв’язок рівнів Spring-проекту

7.2 Модуль роботи зі сховищами даних

Повну діаграму класів модулю роботи зі сховищами даних наведено у додатку Д. Призначення цього модулю – взаємодія зі сховищами даних, які призначені для збереження датасетів. На рисунку 7.12 наведено діаграму класів, які пов’язані у єдиний інтерфейс доступу до сховищ даних.

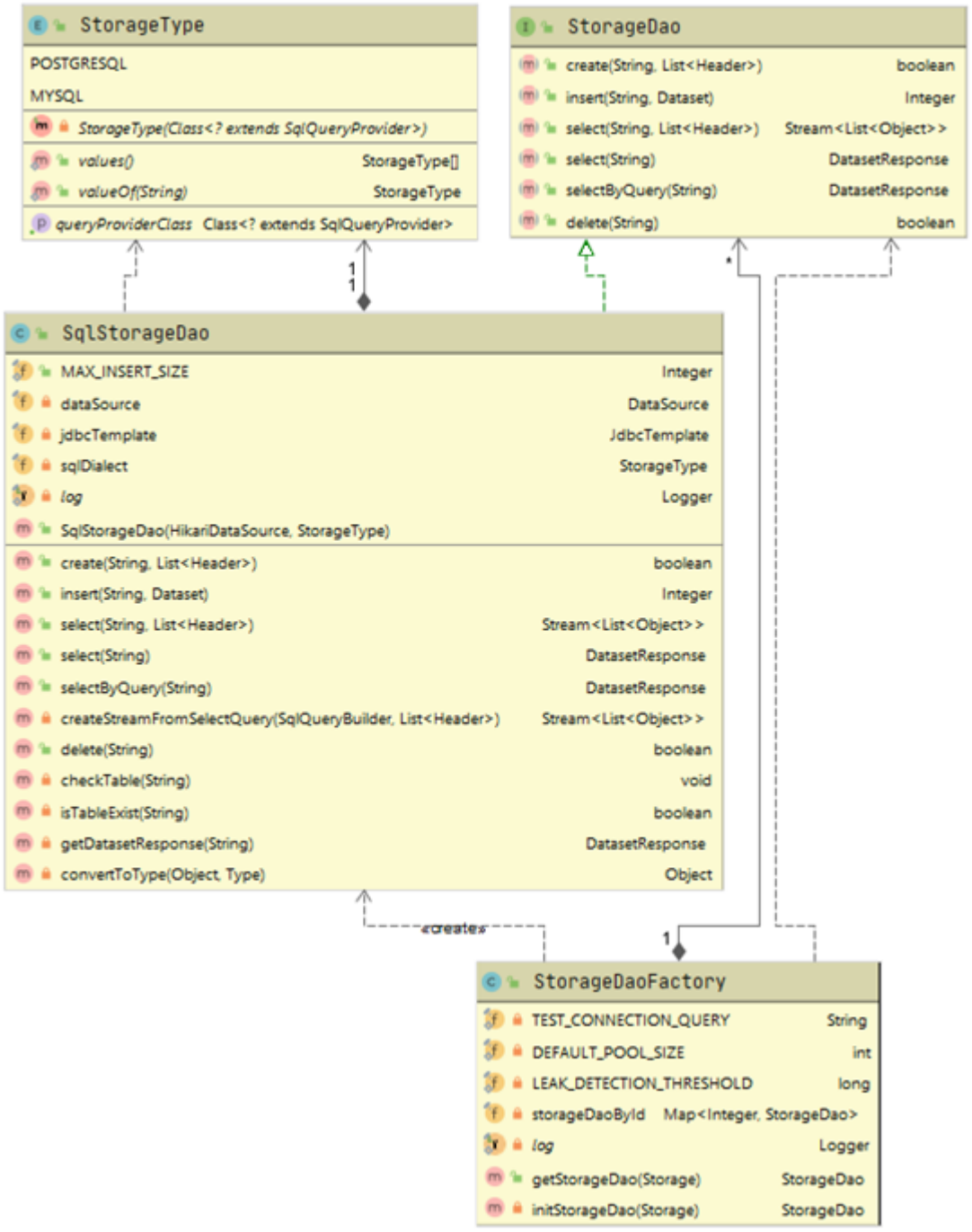

Рисунок 7.12 – Діаграма класів інтерфейсу доступу до сховищ даних

Модуль завантаження даних, як і модуль роботи із джерелами даних, взаємодіє з БД, але останній обмежується лише операцією читання. Окрім цієї операції, модуль завантаження даних надає інтерфейс для виконання операцій створення й видалення об’єктів БД, та запису й видалення даних. Інтерфейс StorageDao містить методи, які відповідають наведеним операціям. У таблиці 7.2 наведено методи інтерфейсу та відповідні їм SQL-оператори.

Таблиця 7.2 – Методи інтерфейсу StorageDao

Метод	Аргументи	Призначення	SQL-оператор
create(String, List<Header>)	Ім’я датасету, список колонок.	Створює таблицю датасету у сховищі.	CREATE
insert(String, Dataset)	Ім’я датасету, датасет.	Додає дані у таблицю датасету. Якщо таблиці не існує, викликається метод create.	INSERT
select(String, List<Header>)	Ім’я датасету, список колонок.	Виконує вибірку даних на основі необхідних колонок.	
select(String)	Ім’я датасету.	Виконує вибірку даних з усіх колонок датасету.	
selectByQuery(String)	SQL-запит.	Виконує вибірку даних за попередньо сформованим запитом. При обробці запиту природньою мовою генерується SQL-запит, який виконується за допомогою даного методу.	
delete(String)	Ім’я датасету.	Видаляє таблицю датасету зі сховища.	DROP

Даний інтерфейс розроблено за принципом шаблону проектування DAO, тобто існує абстрактний інтерфейс, який надає доступ до БД. У якості реалізації даного інтерфейсу виступає клас SqlStorageDao, який інкапсулює механізми взаємодії із реляційними БД. Таким чином, у систему закладено можливість масштабування шляхом додавання нових типів сховищ даних, наприклад NoSQL БД, взаємодія із якими відрізняється від реляційних БД, що вимагає розробки власної імплементації StorageDao.

У розробленій системі у якості сховищ даних обрано дві реляційні БД – MySQL та PostgreSQL. Кожному типу сховища відповідає константа с перечисления

StorageType, яке містить єдине поле – queryProviderClass, яке містить клас-постачальник специфічних запитів для конкретного діалекту SQL.

Екземпляр класу SqlStorageDao створюється за допомогою єдиного конструктора, який приймає два аргументи:

– HikariDataSource, який містить пул об’єктів Connection для взаємодії зі сховищем;

– StorageType, що визначає тип сховища даних, із яким буде взаємодіяти даний екземпляр.

Для отримання об’єктів StorageDao використано породжуючий шаблон проектування – фабричний метод. Клас StorageDaoFactory є компонентом Spring та за замовчуванням життєвий цикл цього об’єкта – сінглтон. Даний клас має єдиний публічний метод – getStorageDao, який на основі об’єкту Storage, який отримується із БД, повертає екземпляр StorageDao. Фабричний метод, що надає інтерфейс створення екземплярів, розроблено таким чином, щоб кожній сутності сховища відповідав лише один екземпляр StorageDao, оскільки він містить пул з’єднань із БД і треба уникати випадків створення цих об’єктів, тому що вони займають вагому частину ресурсів. Для цього створено поле storageDaoById, що представляє собою колекцію Map, яка містить ідентифікатори сховищ та відповідні їм екземпляри StorageDao. У випадку, коли у колекції немає екземпляру StorageDao для сховища, виконується його ініціалізація. Створюється об’єкт HikariDataSource, який на основі параметрів підключення створює пул з’єднань. Стандартний розмір пулу – 12. Наступник кроком ініціалізується об’єкт StorageDao, який зберігається у колекцію та повертається у якості результату методу. У випадку, коли колекція містить відповідний об’єкт для ідентифікатора сховища, ініціалізація не виконується і об’єкт одразу повертається.

Для запобігання ситуацій, коли декілька потоків звертаються до одного і того ж ресурсу (у даному випадку – колекція Map), виконано синхронізацію на рівні об’єкту при зверненні до колекції. На рисунку 7.13 наведено фрагмент коду, який демонструє, як само імплементовано синхронізацію та принцип роботи фабричного методу.

```
public StorageDao getStorageDao(Storage storage) {
    if (!storageDaoById.containsKey(storage.getId())) {
        synchronized (this) {
            if (!storageDaoById.containsKey(storage.getId())) {
                var storageDao : StorageDao = initStorageDao(storage);
                storageDaoById.put(storage.getId(), storageDao);
                return storageDao;
            }
        }
    }

    return storageDaoById.get(storage.getId());
}
```

Рисунок 7.13 – Фрагмент коду фабричного методу

Після першого звертання до колекції, у випадку, коли відповідний екземпляр StorageDao не знайдено, виконується синхронізація на рівні об’єкту самого екземпляру, що виконує даний метод. Даний механізм гарантує, що тільки один потік зможе виконувати даний блок коду для даного екземпляру класу. Оскільки фреймворк Spring гарантує, що екземпляр класу StorageDaoFactory буде єдиним об’єктом у системі, то синхронізацію виконано саме на рівні об’єкту замість синхронізації на рівні класу.

Алгоритм роботи кожного метод класу SqlStorageDao, незалежно від того, яка операція виконується зі сховищем даних, складається з наступних кроків:

– побудова SQL-запиту на основі вхідних даних;

– виконання SQL-запиту за допомогою класу JdbcTemplate – компоненту фреймворку Spring;

– отримання результату виконання SQL-запиту.

На рисунку 7.14 наведено діаграму класів, що демонструє зв’язок між імплементаціями інтерфейсів StorageDao та SqlQueryBuilder.

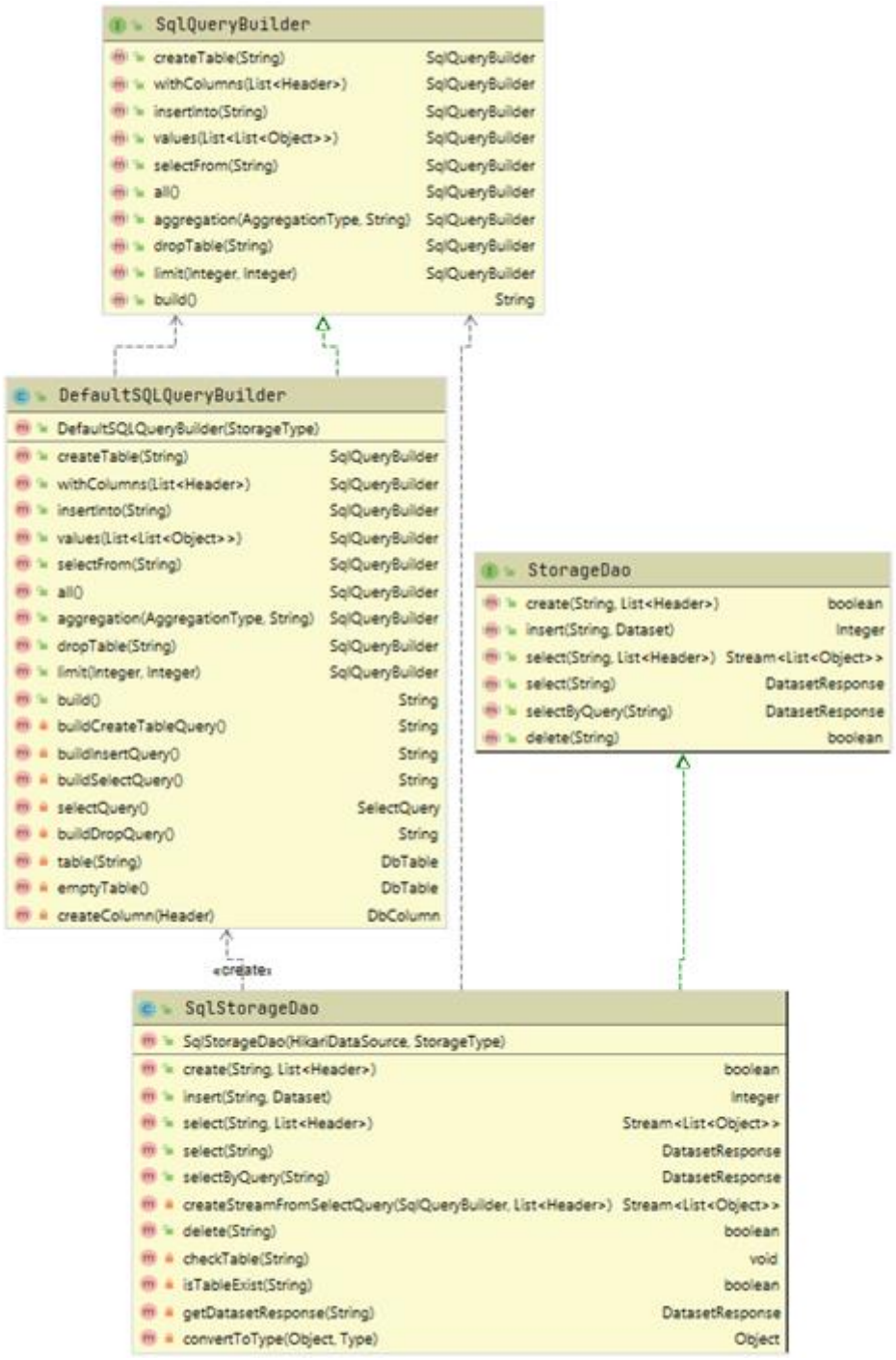

Рисунок 7.14 – Діаграма класів, що демонструє зв’язок між імплементаціями StorageDao та SqlQueryBuilder

Для побудови SQL-запитів для специфічного типу сховища даних розроблено інтерфейс SqlQueryBuilder та відповідний клас-реалізацію DefaultSqlQueryBuilder. Останній є обгорткою над бібліотекою SqlBuilder, який надає спрощений інтерфейс побудови запитів, реалізований за принципом породжуючого шаблону проектування

Builder. Таким чином, логіку взаємодії із використаною бібліотекою повністю інкапсульовано у класі DefaultSqlQueryBuilder.

Бібліотека SqlBuilder не покриває усі необхідні функціональні можливості, що необхідні для реалізації бізнес-логіки системи, тому додатково було розроблено пакет queryprovider, який надає додатковий функціонал, що використовується у класі DefaultSqlQueryBuilder. На рисунку 7.15 наведено діаграму класів даного пакету.

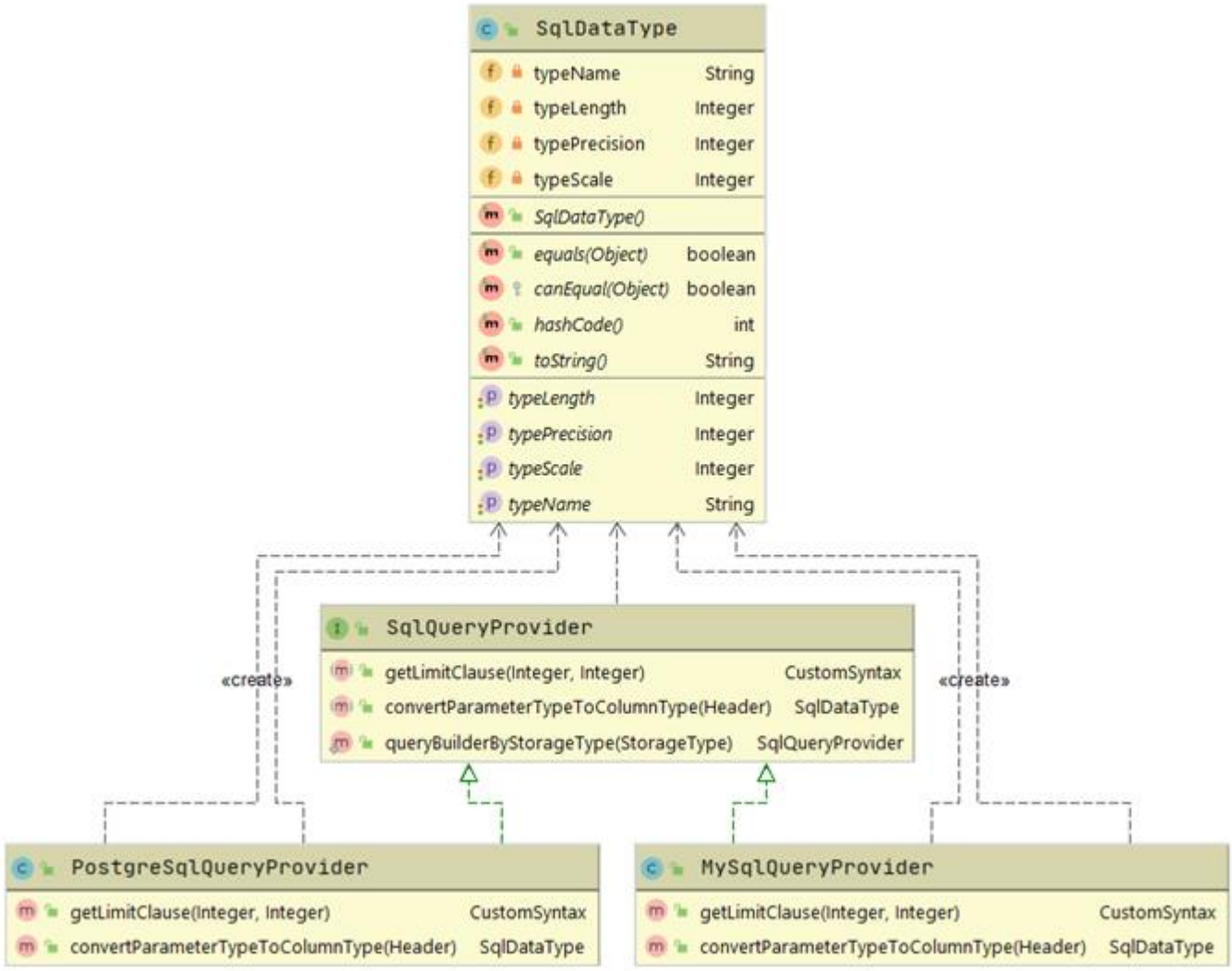

Рисунок 7.15 – Діаграма класів пакету queryprovider

Однією з основних відмінностей між сховищами даних є типи колонок, які використовуються при побудові датасетів. У таблиці 7.3 наведено відповідність між типами, які підтримуються розробленою системою аналізу бізнес-даних, та типами у

сховищах даних (MySQL та PostgreSQL). Типам даних системи відповідають константи перечислення Type, що розроблено у модулі роботи із джерелами даних.

Таблиця 7.3 – Відповідність між типами даних у системі та сховищами даних

Система аналізу бізнес-даних	MySQL	PostgreSQL
DATE	DATETIME	timestamp without time zone
DECIMAL	DOUBLE	double precision
INTEGER	BIGINT	bigint
TEXT	TEXT	text

Таким чином, для цього розроблено інтерфейс SqlQueryProvider, який має дві реалізації, що відповідають кожному підтримуваному у системі сховищу даних. Метод convertParameterTypeToColumnType виконує конвертацію типу даних системи до типу даних сховища, та повертає інформацію про тип, що інкапсульовано у сутності SqlDataType.

Також інтерфейс SqlQueryProvider містить статичний метод queryBuilderByStorageType, який виконує роль фабричного методу. В залежності від значення типу сховища, за допомогою рефлексії створюється екземпляр класу відповідного постачальника запитів SQL. Тип класу постачальника міститься у полі queryProviderClass перечислення StorageType, який визначається у конструкторі (рисунок 7.16), тобто при оголошенні нового типу сховища, треба визначити й відповідний йому клас-постачальник запитів.

Наявність методу getLimitClause зумовлена тим, що різним типам БД відповідають різні конструкції з бібліотеки SqlBuilder. У випадку з SQL оператором LIMIT, для MySQL – це клас MysLimitClause, для PostgreSQL – клас PgLimitClause.

```
public enum StorageType {
    POSTGRESQL(PostgreSqlQueryProvider.class),
    MYSQL(MySqlQueryProvider.class);

    private final Class<? extends SqlQueryProvider> queryProviderClass;
}
```

Рисунок 7.16 – Фрагмент коду перечислення StorageType

Для реалізації сценарію використання завантаження даних у сховище із джерела даних, розроблено сервіс DatasetService, який містить метод upload, що інкапсулює бізнес-логіку, яка пов’язує роботу двох модулів. При завантаженні датасету у систему виконуються наступні кроки:

– отримання датасету із джерела даних;

– отримання сутності потрібного сховища даних, на основі якого отримується об’єкт StorageDao;

– видалення існуючого датасету – якщо сховище містить датасет з потрібним ідентифікатором, то він видаляється;

– запис датасету у сховище.

Для виконання завантаження датасету розроблено відповідний контроллер, що містить REST-запит, при виконанні якого використовується описаний вище сервіс. Окрім запиту на завантаження даних, контролер містить запити для видалення датасету та для отримання даних з нього.

7.3 Модуль представлення даних

Повну діаграму класів модулю представлення даних наведено у додатку Е.

До модулю представлення даних відноситься все, що стосується взаємодії користувача із системою аналізу бізнес-даних шляхом відправки запитів природною мовою за допомогою месенджера Slack, аналізу запиту за допомогою інструментів NLP та візуалізації даних.

У контролері DatasetController створено метод query, який відповідає HTTP запиту, що на вході у тілі запиту приймає запит природною мовою, та у результаті

повертає об’єкт Dataset. Контролер у свою чергу використовує сервіс датасетів, а саме метод queryByNL. За допомогою сервісу NlpService отримується пара об’єктів – SQL-запит та сутність Storage, у якому зберігається потрібний датасет. DatasetService за допомогою фабрики StorageDaoFactory отримує екземпляр StorageDao, та викликає метод selectByQuery, який повертає об’єкт датасету, що і є результатом виконання HTTP запиту. На рисунку 7.17 наведено діаграму класів, що використовуються при обробці запиту природньою мовою.

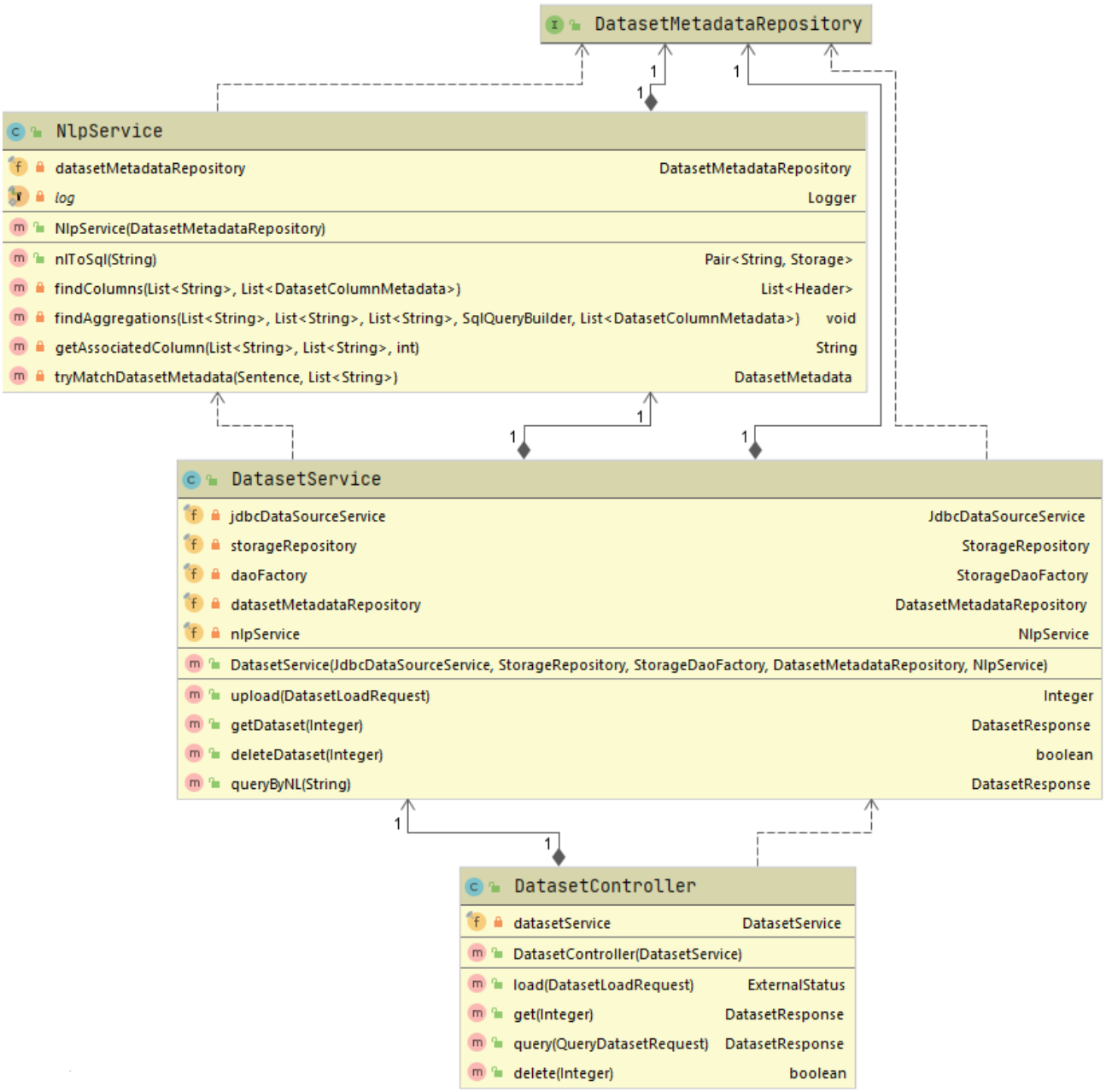

Рисунок 7.17 – Діаграма класів, які використовуються при обробці запиту природньою мовою

Для взаємодії системи аналізу бізнес-даних із месенджером Slack розроблено окремий сервіс Slackbot, який взаємодіє із основним додатком за допомогою REST API та використовує бібліотеку JBot для обміну даними із Slack-сервером. Також даний сервіс виконує функцію представлення даних у зручному для користувача вигляді за допомогою бібліотеки JFreeChart, яка виконує побудову діаграм. Таким чином, сервіс Slackbot виконує наступні функціональні можливості:

– обмін повідомленнями із ботом;

– формування запиту до основного сервісу системи;

– формування візуалізації на основі отриманого датасету (текстова візуалізація або побудова діаграм).

Діаграму класів сервісу Slackbot наведено на рисунку 7.18.

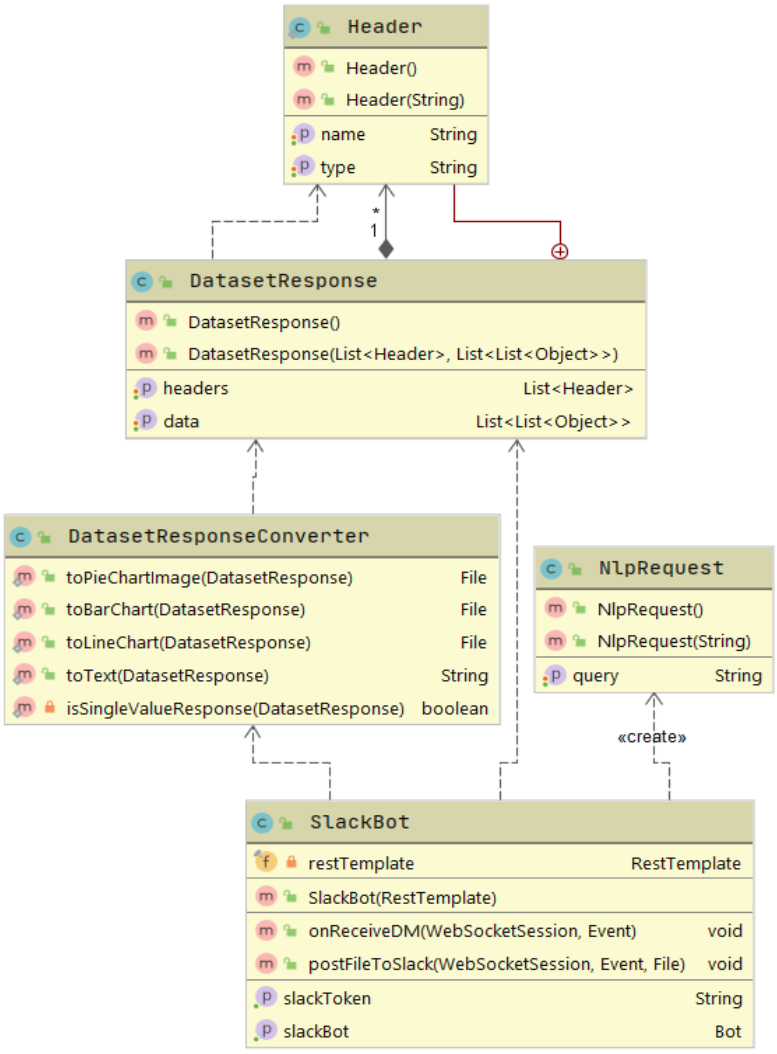

Рисунок 7.18 – Діаграма класів Slackbot сервісу

Центральним класом є SlackBot, для якого батьківським класом є Bot з бібліотеки JBot, що містить необхідні методи для комунікації із Slack API. Метод onReceiveDM представляє собою обробник події, що реагує на прямі повідомлення до боту. Даний метод містить анотацію Controller, що містить властивість events. З рисунку 7.19 видно, що подіями, на які спрацьовує метод onReceiveDM – це отримання прямого повідомлення, або отримання повідомлення при прямому посиланні до бота у каналі. У останньому випадку бот має бути присутнім у каналі, у якому виконується звернення до нього.

```
@Controller(events = {EventType.DIRECT_MENTION, EventType.DIRECT_MESSAGE})
public void onReceiveDM(WebSocketSession session, Event event) {
```

Рисунок 7.19 – Фрагмент коду метода обробника повідомлень

Класи NlpRequest та DatasetResponse описують об’єкти запиту та відповіді при взаємодії із основною частиною системи за допомогою HTTP. Клас DatasetResponseConverter інкапсулює логіку побудови повідомлень на основі датасету. В залежності від запиту, формується текстове повідомлення або відповідь у вигляді картинки формату PNG, що містить потрібну діаграму.

Для прив’язки конкретного боту з воркспейсу Slack до системи аналізу бізнес-даних достатньо додати значення токену API, що унікально ідентифікує бота, у поле slackBotToken у файлі конфігурації Spring Boot додатку application.properties.

Блок-схему алгоритму оброблення запиту природною мовою наведено у додатку Ж. При отриманні повідомлення від користувача сервіс Slackbot формує запит до основного модулю системи на отримання датасету на основі запиту природною мовою. У об’єкті NlpService виконується основна логіка роботи алгоритму, єдиним публічним методом якого є метод nlToSql. Першим кроком, виконується токенізація та лематизація речення. Для токенізації використовується технологія POS-tagging, яка допомагає визначити частину мови для токенів (слів). Процес лематизації – приведення слова до словникової форми, потрібен для

встановлення співвідношення токенів запиту із назвами таблиць та колонок, оскільки вони за стандартом іменування знаходяться у словниковій формі.

На основі отриманих даних виконується встановлення співвідношення токенів між атрибутами таблиці та параметрами запиту. Першим кроком алгоритм намагається встановити назву датасету, з якого виконується отримання даних. За це відповідає метод tryMatchDatasetMetadata. У випадку, якщо потрібного датасету не знайдено – генерується повідомлення про помилку. Наступним кроком система намагається встановити за допомогою окремого методу findAggregations, чи потрібно використовувати функції агрегування при формуванні SQL запиту. При цьому відбувається ітерація по всім синонімам для кожного типу агрегації, та визначається, чи є співвідношення між ними та токенами запиту. Синоніми агрегацій визначаються прямо у конструкторі перечислення AggregationType (рисунок 7.20). При всіх порівняннях строк не враховується регістр літер.

```
COUNT( ...synonyms: "count", "number"),
SUM( ...synonyms: "sum", "total"),
MAX( ...synonyms: "maximum", "highest", "most", "largest"),
MIN( ...synonyms: "minimum", "least", "lowest", "smallest", "minimal"),
AVG( ...synonyms: "average", "mean");
```

Рисунок 7.20 – Фрагмент коду перечислення AggregationType

Після цього система визначає колонки, які потрібні при виконанні запиту. Якщо при виконанні алгоритму встановлено і функцію агрегування, і додаткові колонки, то SqlQueryBuilder автоматично додасть групування по колонкам, як того вимагає мова SQL. При встановленні атрибутів SQL-запиту перевіряються лише ті токени, які за тегами POS-tagging є іменниками.

Наступним кроком будується та виконується SQL-запит, за результатом якого формується об’єкт датасету, що передається сервісу Slackbot. Сервіс аналізує, чи є у запиті згадування про тип діаграми, та на основі цього формує текстове повідомлення або будує діаграму. У останньому випадку виконується завантаження

картинки у якості повідомлення-відповіді. У інших випадках – відправляється повідомлення у вигляді тексту.

7.4 Діаграма розгортання системи

Діаграму розгортання системи аналізу бізнес-даних наведено у додатку И. На діаграмі присутні три вузли пристрої:

- сервер системи;
- HTTP-клієнт для адміністрування;
- пристрій користувача.

Сервер системи складається з трьох вузлів середи виконання. Перший вузол – це чатбот, сервіс, який взаємодіє із зовнішнім slack-сервером та із модулем інтеграції даних. Slack-сервер у свою чергу взаємодіє із пристроєм користувача. Наступний вузол – це модуль інтеграції даних, що виконує функції по взаємодії із джерелами та сховищами даних та оброблення запитів природною мовою. Останній вузол – це БД PostgreSQL, яка використовується системою для збереження інформації та виступає у ролі стандартного сховища даних.

8 ТЕСТУВАННЯ СИСТЕМИ

Для підтвердження працездатності системи було проведено ручне тестування, що виконувалось у двох середовищах. Перше середовище – це додаток PostMan, що дозволяє тестувати API програмного забезпечення. Друге середовище – платформа обміну повідомленнями, тобто месенджер Slack.

Для тестування системи було обрано набір даних – щоденні продажі компанії (Daily sales), що містить інформацію про збут товарів компанії. Детальний опис датасету наведено у таблиці 8.1.

Таблиця 8.1 – Опис набору даних

Колонка	Призначення
Date	Дата продажів
Country	Країна продажів
Channel	Канал (спосіб) збуту продукції
Product	Назва продукту
Product category	Категорія продукту
Product subcategory	Підкатегорія продукту
Units	Обсяг реалізації товару (кількість)
Sales	Продаж (кількість грошей)
Cost	Собівартість проданого товару
Profit	Прибуток

Джерелом даних виступає стороння БД MySQL, профіль підключення до якої заздалегідь збережено у БД, що взаємодіє із додатком, у таблиці jdbc_data_source. Перевірка підключення до джерела даних виконується HTTP запитом, що у тілі запиту приймає ідентифікатор джерела даних. Результат виконання запиту представлено на рисунку 8.1. У тілі відповіді отримується інформація про БД (назва, версія і т.д.).

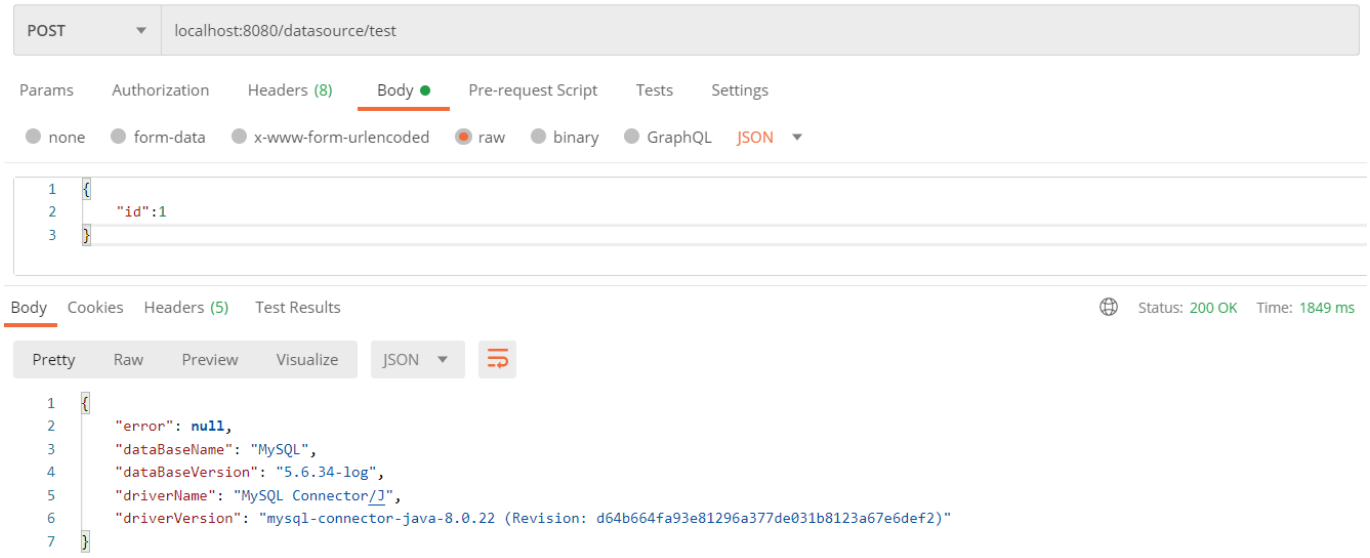

Рисунок 8.1 – Результат виконання запиту перевірки підключення до джерела даних

У якості сховища даних було використано окрему БД PostgreSQL та збережено інформацію для підключення до сховища у таблиці storage. На рисунку 8.2 наведено результат виконання запиту завантаження даних у сховище. У тілі запиту наведено ідентифікатори датасету, джерела та сховищ даних, та SQL-запит, на основі якого генерується датасет.

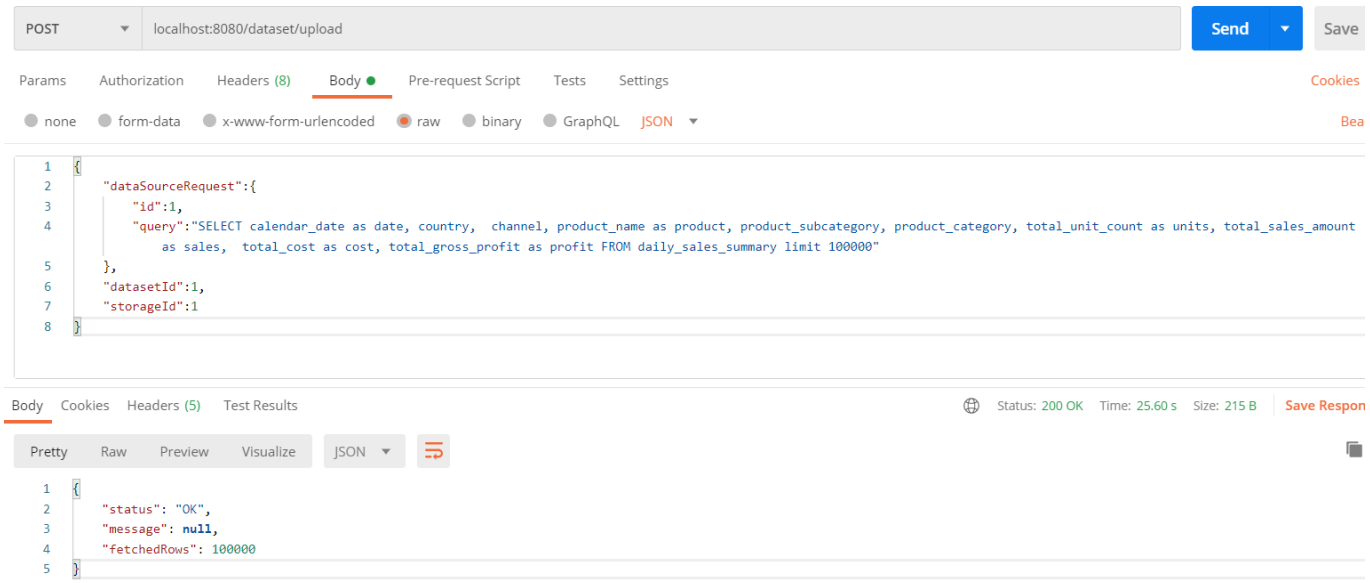

Рисунок 8.2 – Результат виконання запиту завантаження даних у сховище

Як видно зі скріншоту, проведено зчитування та запис ста тисячі рядків, на що система видратила приблизно двадцять п’ять секунд. Така затримка зумовлена тим, що фізично джерело та сховище даних знахотяться у різних підмережах. Таким чином, виконано запис даних із бази MySQL до бази PostgreSQL. Фрагмент візуалізації із таблиці сховища даних наведено на рисунку 8.3.

	date timestamp wit	country text	channel text	product text	product_subcat text	product_categor text	units bigint	sales double precision	cost double precision	profit double precision
1	2018-12-16 ...	Australia	corporate sales	2009 Spring...	red wine	wine	118	1180	594.72	585.28
2	2018-12-16 ...	Canada	corporate sales	2009 Spring...	red wine	wine	268	2680	1358.76	1321.24
3	2018-12-16 ...	France	corporate sales	2009 Spring...	red wine	wine	422	4220	2139.54	2080.46
4	2018-12-16 ...	Germany	corporate sales	2009 Spring...	red wine	wine	33	330	167.97	162.03
5	2018-12-16 ...	Russia	corporate sales	2009 Spring...	red wine	wine	132	1320	669.24	650.76
6	2018-12-16 ...	Spain	corporate sales	2009 Spring...	red wine	wine	33	330	166.32	163.68
7	2018-12-16 ...	United Kingdom	corporate sales	2009 Spring...	red wine	wine	170	1700	861.9	838.1
8	2018-12-16 ...	United States	corporate sales	2009 Spring...	red wine	wine	433	4330	2182.32	2147.68
9	2018-12-16 ...	Australia	e-mail marketing	2009 Spring...	red wine	wine	171	1710	861.84	848.16
10	2018-12-16 ...	Canada	e-mail marketing	2009 Spring...	red wine	wine	271	2710	1373.97	1336.03

Рисунок 8.3 – Фрагмент даних з таблиці датасету

Для тестування у месенжері, створено окремий воркспейс та бота, що має назву data-analysis-bot. Оскільки назву датасету є Daily sales, вона має фігурувати у кожному запиті, як ідентифікатор датасету. Важливим показником при аналізі даних є визначення кількості унікальних значень певного атрибуту. Виконання запиту на отримання кількості унікальних країн, які фігурують у наборі даних, зображено на рисунку 8.4.

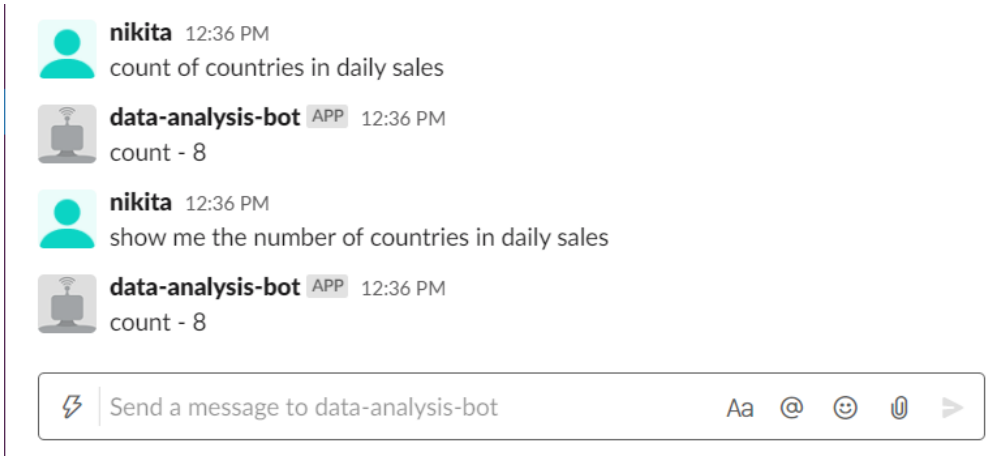

Рисунок 8.4 – Результат виконання запиту на отримання кількості унікальних країн природною мовою

На скріншоті зображено два різних формулювання запитів природньою мовою, проте результат, що видає система, однаковий.

На рисунку 8.5 зображено результат виконання відповідного SQL-запиту. Як видно, результати виконання запитів природною мовою та мовою SQL співпадають.

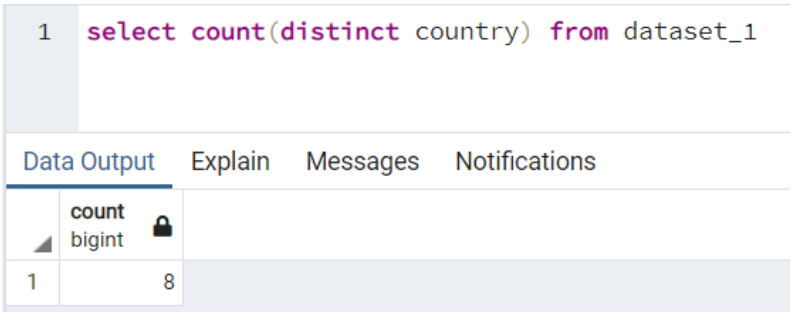

Рисунок 8.5 – Результат виконання SQL-запиту на отримання кількості унікальних країн

Аналогічно до вище наведеного прикладу, продестовано інші способи використання функцій агрегації:

- min (рисунок 8.6);
- max (рисунок 8.7);
- sum (рисунок 8.8);
- avg (рисунок 8.9).

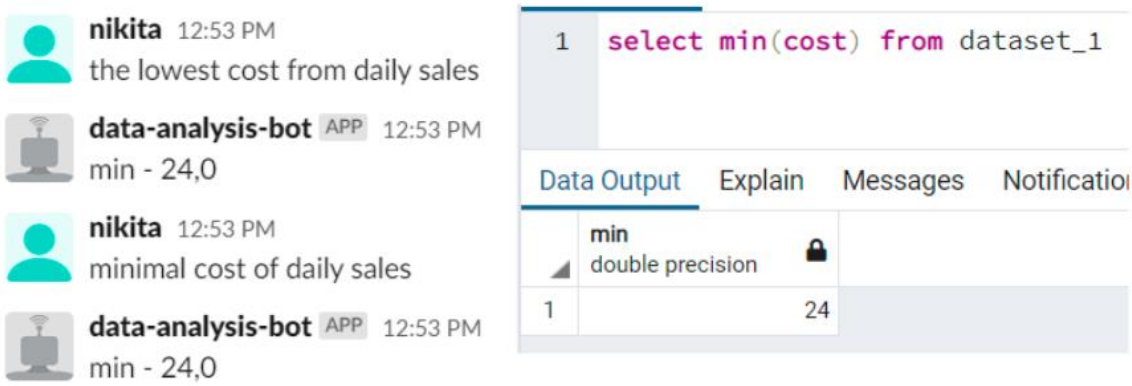

Рисунок 8.6 – Результат виконання запиту на отримання мінімального значення собівартості проданих товарів

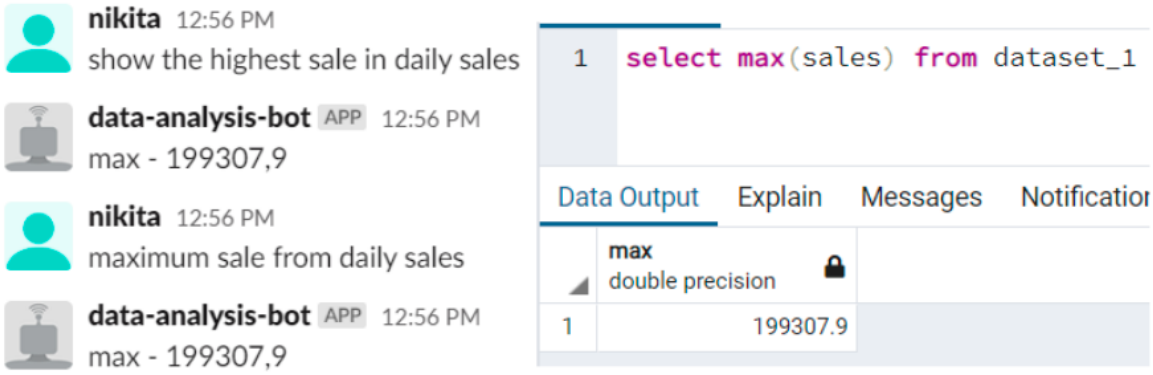

Рисунок 8.7 – Результат виконання запиту на отримання максимального значення продажів

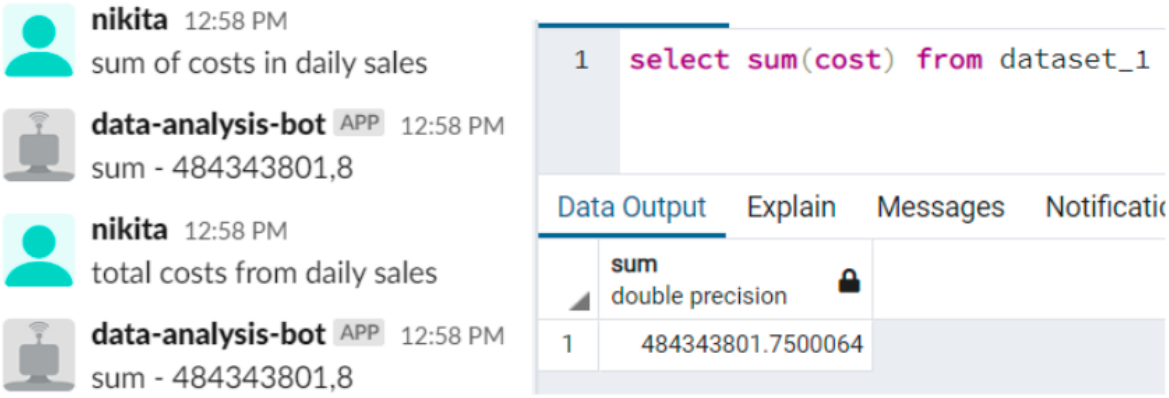

Рисунок 8.7 – Результат виконання запиту на отримання суми собівартостей продажів за всі дні

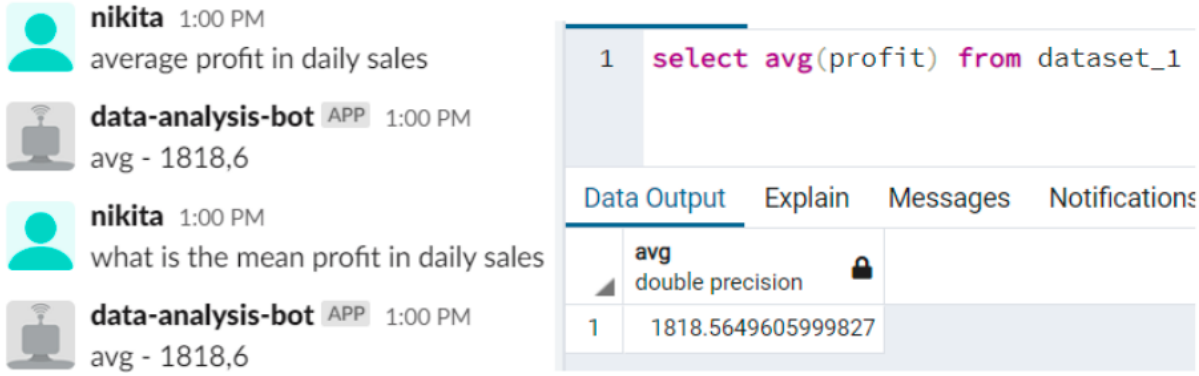

Рисунок 8.8 – Результат виконання запиту на отримання середнього значення прибутку

Наступним важливим елементом у аналізі бізнес-даних є визначення певних метрик, що необхідні для визначення стану бізнесу. На рисунку 8.9 наведено приклад виконання запиту на отримання суми продажів за усі дні в залежності від країни.

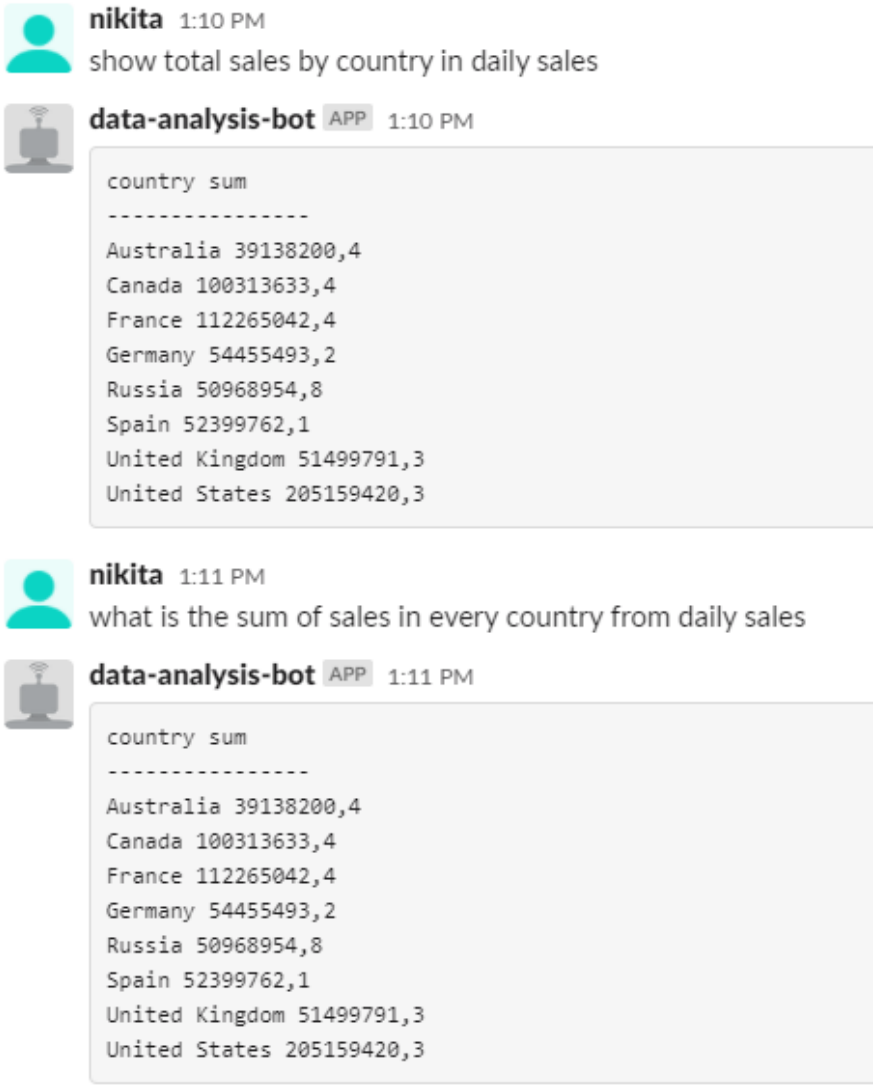

Рисунок 8.9 – Результат виконання запиту природною мовою на отримання суми продажів в залежності від країни

Результат виконання відповідного SQL-запиту наведено на рисунку 8.10.

```
1  select country, sum(sales) from dataset_1 group by 1
```

Data Output | Explain | Messages | Notifications

	country text	sum double precision
1	United King...	51499791.310000055
2	Spain	52399762.13000022
3	Australia	39138200.40999995
4	Germany	54455493.15000006
5	Russia	50968954.8
6	Canada	100313633.38000025
7	United States	205159420.2700002
8	France	112265042.35999961

Рисунок 8.10 – Результат виконання SQL-запиту на отримання суми продажів в залежності від країни

Для більш зрозумілого представлення даних, користувач має можливість вказати тип діаграми, у вигляді якої очікується результат виконання запиту природньою мовою. На рисунку 8.11 зображено приклад використання такого запиту у вигляді кругової діаграми.

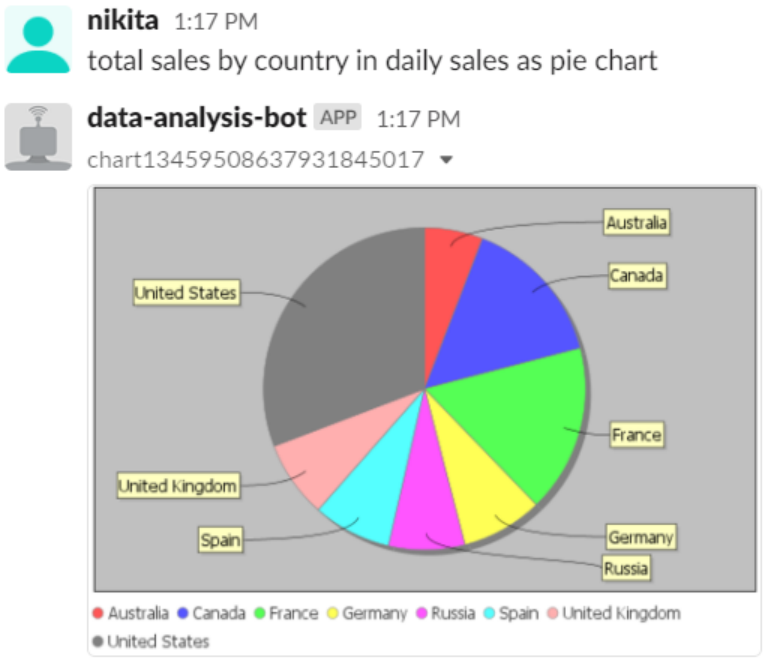

Рисунок 8.11 – Результат виконання запиту природньою мовою у вигляді діаграми

9 СТАРТАП-ПРОЕКТ

9.1 Опис ідеї проекту

У магістерській дисертації розроблено систему для аналізу бізнес-даних з використанням технологій обробки природних мов. Система дозволяє:

- спростити взаємодію «користувач-система»;
- працювати з даними без спеціальних навичок з області IT;
- обробляти дані з різних джерел;
- вилучати дані з необхідного джерела без прямого його вказання у запиті.

Зміст ідеї, основні вигоди для користувача, можливі аспекти застосування стартап- проекту наведено у таблиці 9.1.

Таблиця 9.1 – Опис ідеї стартап-проекту

Зміст ідеї	Напрямки застосування	Вигоди для користувача
Створення системи для аналізу бізнес-даних з використанням технологій обробки природних мов та агрегацією різних джерел даних в єдиний формат	Бізнес аналітика	Користувач отримує вигоду в двох аспектах: взаємодія з представленням даних та взаємодія з самими даними. Вигоди у взаємодії с представленням даних: - спрощення взаємодії з інтерфейсом системи; - вирішення проблеми нестачі спеціалістів по роботі з даними, трактуванням результатів BI. Вигоди у взаємодії с даними: - обробка даних з різних джерел, приведенням їх до єдиного формату; - можливість вилучати дані з необхідного сховища без прямого його вказання.

Системи бізнес-аналізу з використанням обробки природних мов – нова тенденція в BI 2020 року. Аналоги до розробленого продукту вже існують, проте він має певні переваги і інноваційні рішення. У таблиці 9.2 проведено аналіз техніко-економічний переваг проекту, а саме його слабких, нейтральних та сильних сторін у порівнянні з конкурентами.

Таблиця 9.2 – Визначення сильних, слабких та нейтральних характеристик ідеї проекту

п/п	Техніко-економічні характеристики ідеї	(потенційні) товари/концепції конкурентів				W (слабка сторона)	N (нейтральна сторона)	S (сильна сторона)
		Мій проект	QlikView	Tableau	Power BI			
1.	Взаємодія з даними з використанням технології обробки природних мов	Основний фокус системи – NLP	NLP використовується як додаткова функція	NLP використовується як додаткова функція	NLP використовується як додаткова функція			
2.	Взаємодія за допомогою чат-боту	Так	Ні	Так	Так			
3.	Агрегація різних джерел даних до єдиного формату	Так	Так	Так	Так			

4.	Робота з даними без необхідності вказання їх сховища, можливість використання декількох сховищ	Так	Ні	Ні	Ні			

За отриманими результатами аналізу техніко-економічних характеристик ідеї, їх сильних сторін, можна стверджувати, що проект є конкурентноспроможним на сучасному ринку рішень для бізнес-аналітики.

9.2 Технологічний аудит ідеї проекту

Для реалізації системи аналізу бізнес даних з використанням технологій обробки приробних мов необхідно визначитися з інструментами для таких аспектів: інтеграція даних, адміністрування даних та інтеграція з месенджером. Аналіз складових технологічної здійсненності ідеї проекту наведено в таблиці 9.3.

Таблиця 9.3 – Технологічна здійсненність ідеї проекту

п/п	Ідея проекту	Технології її реалізації	Наявність технологій	Доступність технологій
1.	Інтеграція даних	JDBC	Наявна	Доступна
Обрана технологія реалізації ідеї проекту: JDBC				
		Java	Наявна	Доступна

		Python	Наявна	Доступна
		PHP	Наявна	Доступна
Обрана технологія реалізації ідеї проекту: Java				
		Slack API	Наявна	Доступна
		Telegram API	Наявна	Доступна
Обрана технологія реалізації ідеї проекту: Slack API				

Технологічна реалізація системи аналізу бізнес даних з використанням технологій обробки природних мов можлива. Більш того для реалізації більшості аспектів ідеї стартап-проекту існує декілька наявних і доступних засобів. У силу їх переваг обрано наступні: API для доступу до даних для їх інтегрування – JDBC, мова програмування для адміністрування даних – Java, месенджер для взаємодії з користувачем – Slack і відповідно Slack API для інтеграції системи з ним.

9.3 Аналіз ринкових можливостей запуску стартап-проекту

Для планування напрямків впровадження проекту у обраному середовищі необхідно визначити можливості та загрози, які можуть виникнути в процесі. Це можна здійснити на основі аналізу потреб клієнтів (потенційних) та вже існуючих на ринку аналогів.

Проведений аналіз попиту занесено до таблиці 9.4 .

Таблиця 9.4 – Попередня характеристика потенційного ринку стартап-проекту

№ п/п	Показники стану ринку (найменування)	Характеристика
1.	Кількість головних гравців, од	20
2.	Загальний обсяг продаж, грн/ум.од	$ 7,5 млрд

3.	Динаміка ринку (якісна оцінка)	Зростає
4.	Наявність обмежень для входу	Високі початкові капіталовкладення
5.	Специфічні вимоги до стандартизації та сертифікації	Українське законодавство не має специфічних вимог для стандартизації такого товару
6.	Середня норма рентабельності в галузі (або по ринку), %	29.9%

Попит на BI системи зростає, особливо з використанням NLP, адже кожного дня створюється велика кількість даних для обробки, а попит на спеціалістів роботи з ними перевищує пропозицію. Вихід на ринок бізнес-аналітики має лише одну перешкоду – високі стартові капіталовкладення, проте рентабельність становить 29.9 %. За отриманими результатами аналізу можна зробити висновки, що ринок є привабливим для входження.

Для системи аналізу бізнес-даних з використанням технологій обробки природних мов потенційними клієнтами є корпорації та компанії будь-якої галузі, які мають необхідність використання інструментів Business Intelligence.

У таблиці 9.5 описано потенційну групу користувачів системи та її можливі специфічні вимоги до системи.

Таблиця 9.5 – Характеристика потенційних клієнтів стартап-проекту

№ п/п	Потреба, що формує ринок	Цільова аудиторія (цільові сегменти ринку)	Відмінності у поведінці різних потенційних цільових груп клієнтів	Вимоги споживачів до товару
1.	Потреба у спрощенні взаємодії між			до системи: наявність зрозумілого

	користувачем та системою	інструменти Business Intelligence для аналізу, обробки та візуалізації даних	презентація; – інтеграція; – налаштування; – мобільність; – вимоги до навчання; – ціноутворення; – підтримка. Формування поведінки динамікою ринку	інтерфейсу з можливістю взаємодії за допомогою природної мови; до компанії-постачальника: доступна цінова політика
2.	Потреба у об’єднанні різнорідних джерел даних			до системи: варіативність налаштувань проекту
3.	Потреба у можливості використання декількох сховищ даних			до системи: гнучкість у конфігурації системи

Після здійснення огляду характеристик потенційних клієнтів, проводиться аналіз факторів сприяння впровадженню спартапу та факторів загроз (таблиці 9.6 й 9.7).

Таблиця 9.6 – Фактори загроз

№ п/п	Фактор	Зміст загрози	Можлива реакція компанії
1.	Зниження доходів потенційних клієнтів	Продукт стає нерентабельним	Введення акційних пропозицій на дешевшу версію продукту з меншою функціональною навантаженістю

2.	Технологічні прориви в рішеннях конкурентів	Методологія та розроблена система може перейти на рівень застарілої при винайденні нових більш результативних рішень в BI	Вдосконалення власних технологічних рішень, об’єднання з компанією конкурента

Таблиця 9.7 – Фактори можливостей

№ п/п	Фактор	Зміст можливості	Можлива реакція компанії
1.	Основний фокус в бізнес-аналітиці на NLP	Застосування обробки природних мов у BI – нове явище 2020 року, яке тільки набуває популярності	Монополізація ринку
2.	Розширення функціоналу	Просування базового функціоналу та пошук нових рішень для вдосконалення	Аналіз систем конкурентів, агрегація вдалих рішень у власну систему
3	Зростання інтересу закордонних інвесторів	Полегшення виходу на локальний ринок	Аналіз необхідності додаткової стандартизації та покращення локалізації

З огляду факторів впровадження можна зробити висновок, що наявні як можливості, так і загрози, проте можливостей на даний момент більше. Надалі проводиться аналіз пропозиції: визначаються загальні риси конкуренції на ринку (таблиця 9.8).

Таблиця 9.8 – Ступеневий аналіз конкуренції на ринку

Особливості конкурентного середовища	В чому проявляється дана характеристика	Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною)
1.Тип конкуренції - монополія	Даний сегмент ринку передбачає, що замінників товару не існує	Шляхом об'єднання окремих відомих інструментів та власної розробки надання унікального функціоналу
2. За рівнем конкурентної боротьби: - міжнаціональний	Міжнаціональний – розробка та впровадження не обмежується рамками якоїсь конкретної території	Оформлення різних локалізацій, які має підтримувати система
3. За галузевою ознакою - внутрішньогалузева	Внутрішньогалузева – система призначена для використання в бізнес аналітиці	Фокус на якісній розробці, збільшенню переваг перед системами конкурентів, які забезпечать зайняття стійкої позиції у ніші
4. Конкуренція за видами товарів: - товарно-видова	Конкуренція з іншими продуктами у сфері Business Intelligence	Необхідно покращувати співвідношення «ціна-якість», щоб товари конкурентів вважалися дорогими або неякісними

5. За характером конкурентних переваг – цінова та нецінова	Нецінова – спочатку визначення доцільності застосування в якості інструменту для впровадження в роботу компанії, цінова – рівень доходів потенційних споживачів досить високий, впровадження системи не потребує великих затрат	Покращення співвідношення «ціна-якість»
6.За інтенсивністю - не марочна	Для виготовлення кінцевого продукту необхідно задіяти сторонні послуги	Диференціація продукту за мотивом отримання найбільш якісного продукту

Далі за моделлю Портера необхідно здійснити детальний аналіз конкуренції в обраній галузі (таблиця 9.9).

Обробка природних мов тільки починає використовуватися в бізнес аналітиці, тому лише невелика частина потенційних конкурентів відслідкувала тенденції і розширила свій функціонал такою можливістю. Проте спрощення взаємодії користувача з системою не основний фокус конкурентів, що значно підвищує шанси впровадження даного стартап-проекту. Замінників системи немає, адже вирішення проблеми різних сховищ даних є інноваційним.

Таблиця 9.9 – Аналіз конкуренції в галузі за М. Портером

Складові аналізу	Прямі конкуренти в галузі	Потенційні конкуренти	Постачальники	Клієнти	Товари-замінники
	QlikView,	Високий бар’єр входження	Постачальники	Корпорації, компанії, які використовують	Замінників немає, у аналогів можливе

Складові аналізу	Tableau, Power BI	через великі початкові капіталовкладення	відсутні	інструменти Business Intelligence для аналізу, обробки та візуалізації даних, мають високий дохід	копіювання інноваційної ідеї
Висновки	Інтенсивна конкуренція	Можливості виходу на ринок є, проте з високим бар'єром	Ні	Отримання якісного продукту з максимально простим інтерфейсом	Замінників немає

У таблиці 9.10 на базі таблиць 9.2, 9.5, 9.6, 9.7 й 9.9 сформовано та обґрунтовано перелік факторів конкурентоспроможності.

Таблиця 9.10 – Обґрунтування факторів конкурентоспроможності

№ п/п	Фактор конкурентоспроможності	Обґрунтування (наведення чинників, що роблять фактор для порівняння конкурентних проектів значущим)
1.	Мінімізація затрат на навчання (роботи з системою)	За рахунок застосування NLP необхідність навчання роботи з системою зникає
2.	Універсальність	Можливість обробки даних з різнорідних джерел, а також можливість використання декількох сховищ даних.
3.	Використання інноваційних рішень та підходів	Використання NLP та агрегування різних джерел та сховищ даних

4.	Простий доступ до даних	Доступ до даних зі смартфону (з месенджеру, чатбот)

Проведено аналіз сильних та слабких сторін стартап-проекту (таблиця 9.11).

Таблиця 9.11 – Порівняльний аналіз сильних та слабких сторін автоматизованої системи моніторингу та управління транспортними викидами

№	Фактор конкурентоспроможності	Бали 1-20	Рейтинг системи-конкурента (Д) у порівнянні з запропонованою						
			-3	-2	-1	-0	+1	+2	+3
1.	Мінімізація затрат на навчання (роботи з системою)	20	Q		T	P			
2.	Універсальність	20			Q, T, P				
3.	Використання інноваційних рішень та підходів	15	Q		T	P			
4.	Простий доступ до даних	18	Q		T	P			

У таблиці 9.11 здійснено порівняння з системами аналогами – QlikView, Tableau, Power BI (в таблиці відповідно Q, T, P).

Заключний етап у визначенні чи можливо провадити стартап – це складання SWOT-аналізу (таблиця 9.12).

Таблиця 9.12 – SWOT- аналіз стартап-проекту

Сильні сторони: інноваційний продукт, який дозволяє спростити доступ до даних та з їх візуальним поданням, забезпечує компанії-клієнту мінімізацію витрат на найм спеціалістів та їх навчання	Слабкі сторони: вузька та специфічна сфера використання

Можливості: зростання інтересу закордонних інвесторів, розширення функціоналу, фокусування в бізнес-аналітиці на NLP, впровадження нового стандарту єдиного формату даних	Загрози: незацікавленість потенційних клієнтів у змінах вже існуючої інфраструктури, поява більш прогресивних конкурентів, зниження доходів клієнтів

Час, за який можливо впровадити систему моніторингу та управління транспортними викидами залежить від можливості отримання ресурсів. Аналіз визначених альтернатив зі сторони строків та імовірності отримання ресурсів наведено у таблиці 9.13.

Таблиця 9.13 – Альтернативи ринкового впровадження стартап-проекту

№ п/п	Альтернатива (орієнтовний комплекс заходів) ринкової поведінки	Ймовірність отримання ресурсів	Строки реалізації
1.	Розробка демо-версії системи, збір та аналіз відгуків	Висока	3 місяці
2.	Розробка системи з повним функціоналом	Висока	1,5 роки
3.	Безкоштовне впровадження продукту в авторитетній компанії певної галузі	Середня	1 рік

Обрано альтернативу розроблення демо-версії системи з метою її впровадження і збору відгуків та досвіду користувачів, адже по термінах реалізації такий комплекс є найкоротшим, а можливість отримання необхідних ресурсів – високою.

9.4 Розроблення ринкової стратегії

У першу чергу, визначено стратегії охоплення ринку та цільові групи споживачів (таблиця 9.14).

Таблиця 9.14 – Вибір цільових груп потенційних споживачів

№ п/п	Опис профілю цільової групи потенційних клієнтів	Готовність споживачів сприйняти продукт	Орієнтовний попит в межах цільової групи	Інтенсивність конкуренції в сегменті	Простота входу у сегмент
1.	Корпорації, компанії, які використовують інструменти Business Intelligence для аналізу, обробки та візуалізації даних	Висока	Високий	Середня	Середня склад-ність

Фокус-група потенційних користувачів одна – компанії, які використовують інструменти Business Intelligence, спрощення взаємодії з користувачем за допомогою NLP тренд 2020 року, тому споживачі готові швидко прийняти систему. На основі аналізу потреб цієї цільової групи визначено стратегію для розвитку ідеї в сегменті. Результати наведено у таблиці 9.15.

Таблиця 9.15 – Визначення базової стратегії розвитку

№ п/п	Обрана альтернатива розвитку проекту	Стратегія охоплення ринку	Ключові конкурентоспроможні позиції відповідно до обраної альтернативи	Базова стратегія розвитку

1.	Розробка демо-версії системи, збір та аналіз відгуків	Концентрованого маркетингу	Простота, універсальність, використання новітніх технологій	Стратегія спеціалізації

За базову стратегію подальшого розвитку взято стратегію спеціалізації через концентрацію на одному сегменті клієнтів. Далі обрано стратегію конкурентної поведінки (таблиця 9.16).

Оскільки система не є першопрохідцем на ринку і деякі лідери бізнес-аналітики у поточному році доповнили свій функціонал обробкою природних мов, в якості стратегії конкурентної поведінки обрано стратегію наслідування лідеру (певного функціоналу, як, наприклад, оформлення результатів). Наступним етапом є визначення стратегії позиціонування на базі вимог користувачів та в залежності від обраної стратегії розвитку (таблиця 9.17).

Таблиця 9.16 – Визначення базової стратегії конкурентної поведінки

п/п	Чи є проект «першопрохідцем» на ринку?	Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?	Чи буде компанія копіювати основні характеристики товару конкурента, і які?	Стратегія конкурентної поведінки
	Ні	Так, так	Так, оформлення результатів	Стратегія наслідування лідеру

Таблиця 9.17 – Визначення стратегії позиціонування

№ п/п	Вимоги до товару цільової аудиторії	Базова стратегія розвитку	Ключові конкурентоспроможні позиції власного стартап-проекту	Вибір асоціацій, які мають сформувати комплексну позицію власного проекту (три ключових)
1.	Наявність зрозумілого інтерфейсу з можливістю взаємодії за допомогою природної мови	Стратегія диференціації	Основний фокус системи – використання технологій обробки природних мов, інтеграція з месенджером	Простота
2.	Варіативність налаштувань проекту	Стратегія диференціації	Налаштування складної інфраструктури	Економічність у використанні апаратних ресурсів
3.	Гнучкість у конфігурації системи	Стратегія диференціації	Можливість використання різних сховищ даних	Універсальність

9.5 Розроблення маркетингової програми стартап-проекту

На базі результатів аналізу конкурентоспроможності системи визначено маркетингову концепцію (таблиця 9.18).

Таблиця 9.18 – Визначення ключових переваг концепції потенційного товару

№ п/п	Потреба	Вигода, яку пропонує товар	Ключові переваги перед конкурентами (існуючі або такі, що потрібно створити
1.	Потреба у спрощенні взаємодії між користувачем та системою	Можливість отримувати результати бізнес-аналізу даних у спрощений спосіб, без спеціальних технічних знань	Інтеграція з месенджером, чат бот, NLP – основний фокус системи
2.	Потреба у об’єднанні різнорідних джерел даних	Сконфігурована система, яка вилучає та приводить дані до єдиного формату	Процес завантаження даних у сховище не обмежений ресурсами сервера (стосовно пам’яті)
3.	Потреба у можливості використання декількох сховищ даних	Можливість використовувати різні сховища даних для різнорідних датасетів	Аналогів не існує

У таблиці 9.19 розроблено трирівневу маркетингову модель товару.

Таблиця 9.19 – Опис трьох рівнів моделі товару

Рівні товару	Сутність та складові
І. Товар за задумом	Система для аналізу бізнес-даних на основі використання технології природних мов

ІІ. Товар у реальному виконанні	Властивості/характеристики	М/Нм	Вр/Тх /Тл/Е/Ор
	1. Швидкість завантаження даних 2. Актуальність інформації 3. Кросплатформеність 4. Масштабованість 5. Зручність користування	1 М 2 Нм 3 М 4 М 5 М	Тл Тл Тх Тл Е
	Якість: тестування, стандарти		
	Пакування: jar/zip		
	Марка: «NLPlanet»		
	До продажу: пакет послуг для BI компаній		
	Після продажу: знижки		
За рахунок чого потенційний товар буде захищено від копіювання: за рахунок захисту інтелектуальної власності			

Наступний етап – визначення меж встановлення ціни на продукт, спираючись на цінову категорію аналогів та рівень доходів потенційних клієнтів (таблиця 9.20).

Таблиця 9.20 – Визначення меж встановлення ціни

№ п/п	Рівень цін на товари-замінники	Рівень цін на товари-аналоги	Рівень доходів цільової групи споживачів	Верхня та нижня межі встановлення ціни на товар/послугу
1.	-	Біля 150 000 грн. на місяць	Залежить від сфери діяльності	100000-150000 грн. на місяць

Наступний крок – визначення оптимальної системи збуту, де має бути визначено спосіб збуту, глибину каналу збуту, необхідність посередників (таблиця 9.21).

Таблиця 9.21 – Формування системи збуту

№ п/п	Специфіка закупівельної поведінки цільових клієнтів	Функції збуту, які має виконувати постачальник товару	Глибина каналу збуту	Оптимальна система збуту
1	В основному бренд для клієнту не має значення, оцінка технічного директору спирається на наданий функціонал ступінь його підходящості, ціну продукту	Встановлення безпосередніх контактів із споживачами та їх підтримка	Канал нульового рівня, оскільки торговий агент – найманий службовець компанії-виробника	Власні засоби продажу – продажі за допомогою торгових менеджерів

Фінальним кроком у розроблені маркетингової є формування концепції маркетингових комунікацій, яка базується на позиціонуванні та специфіці поведінки клієнтів (таблиця 9.22).

Таблиця 9.22 – Концепція маркетингових комунікацій

№ п/п	Специфіка поведінки цільових клієнтів	Канали комунікацій, якими користую-ться цільові клієнти	Ключові позиції, обрані для позиціонування	Завдання рекламного повідомле-ння	Концепція рекламного звернення

1.	Нові тенденції у організації діяльності та динаміка ринку	Формальні та неформальні канали комунікацій (інтернет)	Повністю налаштоване оточення. Гнучка конфігурація проекту на запуск в різних режимах, Універсальність Доступність та простота, мінімізація затрат на навчання (роботи з системою)	Просунення ідеї проекту, орієнтуючись на клієнтів, які використовують Business Intelligence	Позиціонування на основі порівняння товару із товарами конкурентів

Розроблений проект є перспективним для впровадження та має високі шанси на успіх, адже технології обробки природних мов, на яких базується система – нова тенденція у розвитку бізнес аналітичних систем, попит у галузі високий, рентабельність близько 30 %. Комерціалізація проекту можлива. Бар'єром для входження на ринок є лише економічний чинник – високі початкові капіталовкладення. Аналоги до розробленого продукту вже існують, конкуренція присутня, проте система є конкурентноспроможною завдяки ряду сильних сторін – інноваційний продукт, який дозволяє спростити доступ до даних, роботу з ними та з їх візуальним поданням, забезпечує компанії-клієнту мінімізацію витрат на найм спеціалістів та їх навчання.

Для початку діяльності в якості альтернативи впровадження обрано розробку демо-версії системи, збір та аналіз відгуків клієнтів, на основі чого планується подальший розвиток, який є доцільним у даному випадку.

ВИСНОВКИ

У результаті виконання магістерської дисертації було розроблено систему аналізу бізнес-даних з використанням технології оброблення природних мов, яка дозволила спростити взаємодію користувача із програмними засобами BI.

У ході роботи проведено дослідження сучасного ринку систем аналізу бізнес-даних, їх переваг та недоліків, на базі чого сформовано принципи побудови архітектури програмного рішення. Проведено огляд сучасних інструментів з оброблення природних мов та обрано технології для розробки: мова програмування Java, фреймворк для розробки web-додатків Spring, бібліотека з інструментами оброблення природних мов Stanford NLP, бази даних MySQL та PostgreSQL. Для розробки чатбота обрано корпоративний месенджер Slack.

Перед безпосередньою реалізацією системи було розроблено діаграму сценаріїв використання системи та структурну схему, які дозволили на початкових етапах сформувати цілісне уявлення про майбутній продукт. Наступним кроком спроектовано ER-діаграму, де виділено сутності системи та встановлено зв’язки між ними.

Розроблена система складається з трьох основних модулів: модуль роботи із джерелами даних, модуль роботи зі сховищами даних та модуль представлення даних, для кожного з яких створено діаграму класів. Для кращого розуміння роботи системи побудовано блок-схему алгоритму оброблення запиту природною мовою, а для представлення фізичної моделі системи – діаграму розгортання.

Для кінцевого продукту проведено ручне тестування із використанням прикладу набору даних, яке підтвердило коректність роботи системи.

На фінальному етапі розроблено стартап-проект та визначено його основні перспективи та напрямки розвитку.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Системы бизнес-анализа (BI) [Електронний ресурс] – Режим доступу до ресурсу: https://softline.tm/solutions/business-solutions/sistemyi-biznes-analiza-bi

2. Keith D. Foote A Brief History of Natural Language Processing [Електронний ресурс]/ Keith D. Foote// Dataversity. – 2019. – Режим доступу до ресурсу: https://www.dataversity.net/a-brief-history-of-natural-language-processing-nlp/

3. Е.И. Большакова Автоматическая обработка текстов на естественном языке и компьютерная лингвистика: учеб. пособие / Большакова Е.И., Клышинский Э.С., Ландэ Д.В., Носков А.А., Пескова О.В., Ягунова Е.В. — М.: МИЭМ, 2011. — 272 с.

4. Близнюк Б.О. Современные методы обработки естественного языка / Б. О. Близнюк, Л. В. Васильева, И. Д. Стрельников, Д. С. Ткачук // Вісник Харківського національного університету імені В. Н. Каразіна. Мат.моделювання. Інформаційні технології. Автоматизовані системи управління – Х. : ХНУ імені В.Н. Каразіна, 2017. – № 36. – С. 14–26.

5. Biggest Open Problems in Natural Language Processing [Електронний ресурс] – Режим доступу до ресурсу: https://medium.com/sciforce/biggest-open-problems-in-natural-language-processing-7eb101ccfc9#:~:text=The%20main%20challenge%20of%20NLP,%2C%20syntactic%2C%20and%20semantic%20levels.

6. A. White Natural Language Processing in Business Intelligence Software [Електронний ресурс]/ A. White // Izenda. – 2020. – Режим доступу до ресурсу: https://www.izenda.com/natural-language-processing-business-intelligence/

7. Gartner Magic Quadrant & Critical Capabilities [Електронний ресурс] – Режим доступу до ресурсу: https://www.gartner.com/en/research/magic-quadrant

8. QlikView | Система бизнес-анализа [Електронний ресурс] – Режим доступу до ресурсу: http://www.omniway.ua/products/QlikView

9. What is Tableau? Uses and Applications [Електронний ресурс] – Режим доступу до ресурсу: https://www.guru99.com/what-is-tableau.html

10. Руководство по Power BI: начало работы [Електронний ресурс] – Режим доступу до ресурсу: https://habr.com/ru/company/microsoft/blog/427701/

11. Stanford NLP documentation [Електронний ресурс] – Режим доступу до ресурсу: https://stanfordnlp.github.io/CoreNLP/index.html

12. Richard M. Reese Natural Language Processing with Java. Techniques for building machine learning and neural network models for NLP / Richard M. Reese, AshishSingh Bhatia. – B. : «Packt Publishing», 2018. – 318 c.

13. Most popular relational databases [Електронний ресурс] – Режим доступу до ресурсу: https://plumbr.io/blog/io/most-popular-relational-databases

14. MySQL 8.0 Reference Manual [Електронний ресурс] – Режим доступу до ресурсу: https://dev.mysql.com/doc/refman/8.0/en/

15. PostgreSQL: The World's Most Advanced Open Source Relational Database [Електронний ресурс] – Режим доступу до ресурсу: https://www.postgresql.org/

16. E. Evans Domain-Driven design: Tackling Complexity in the Heart of Software / E. Evans. – W. : « Addison-Wesley Professional», 2003. – 560 c.

17. Рогачев С. Обобщенный Model-View-Controller. Повышение качества графического интерфейса пользователя / Рогачев С. // RSDN Magazine. – М. : «Роспечать», 2008. – № 4. – С. 72.